Décembre
2025

numéro
26

1024 Bulletin de la Société informatique de France

COMITÉ DE RÉDACTION

Contact : 1024@socinfo.fr

Société informatique de France
Institut Henri Poincaré, 11 rue Pierre et Marie Curie, 75231 Paris Cedex 05

Prix public : 32 € (adhérents SIF : −30%)
Directeur de la publication : Yves Bertrand ISSN : 2270-1419

Couverture réalisée par l'agence graindecel, https://www.bauwensceline.com.
Composition réalisée par C&F éditions, https://cfeditions.com.
Dessins imaginés et crayonnés par Frédéric Havet,
https://www-sop.inria.fr/members/Frederic.Havet.

Sommaire

RÉCRÉATION

Éditorial

Nous nous retrouvons pour ce numéro 26 de notre bulletin qui fait une grande place à l'enseignement et à la médiation avec, dans chacun des quatre articles concernés, un retour d'expérience très concret, allant de présentations (par le biais de jeux) de concepts importants de l'informatique à des élèves ou des collégiens, jusqu'à une analyse de la situation dans les premières années de l'enseignement supérieur. Évidemment, on observera également dans ce numéro beaucoup de mouvement autour de l'IA et de son utilisation dans différents domaines… et puis, quelques surprises. Allons-y pour notre petit tour d'horizon…

Dans « La grande muraille d'Égypte », Emmanuelle Saillard propose une activité de médiation permettant de suggérer ce qu'est un calcul ou un algorithme parallèle ; le problème consiste à exécuter des instructions en respectant certaines contraintes de placement qui mettent en exergue les avantages et les inconvénients de la parallélisation. Tout ceci accompagné de quelques questions de culture générale informatique, dont la fameuse histoire (légende ?) à l'origine de l'utilisation du mot « *bug* » en informatique[1]. Alain Busser, lui, nous initie aux réseaux de Petri, ce qui peut être fait d'une manière tout à fait ludique comme le montre son expérience en collège dans laquelle il illustre son propos par le calcul de quelques fonctions arithmétiques élémentaires.

Les cibles d'Emmanuelle Saillard et d'Alain Busser sont les élèves des *petites classes*. Ce n'est pas le cas d'Aslı Grimaud et de Pierre Tchounikine qui parlent pour leur part de leur expérience de l'enseignement de l'informatique dans les premières années de l'enseignement supérieur. Aslı Grimaud présente quelques *ressources logicielles libres et* open source *dans l'enseignement de l'informatique en CPGE MP2I et MPI*, et la façon de les utiliser

1. On oubliera aussi vite que possible les tentatives de « transcription » de ce mot en français, comme on a pu oublier certaines velléités *normatives* concernant un vocabulaire français en informatique.

DOI:10.48556/SIF.1024.26.3

pour articuler les différentes phases de l'enseignement d'informatique en classes prépartoires. Pierre Tchounikine, quant à lui, nous parle de *quelques difficultés de l'enseignement de l'algorithmique et de la programmation aux débutants en licence*. Il analyse en profondeur les travers du processus global de l'enseignement de l'informatique et la façon dont les apprenants le reçoivent, appréhendent les nouvelles notions et leur articulation, et réagissent au schéma pédagogique.

Dans son article « BI4people : le décisionnel pour tous », Jérôme Darmont décrit la vie d'un projet ANR autour de l'informatique décisionnelle. Les présupposés et les attendus du projet sont décrits ainsi que son organisation en lots de travaux. Estelle Delouche décrit une utilisation des IA qui devrait permettre d'assister l'analyse des signaux d'un électromyogramme afin de prévenir les lésions du ligament croisé antérieur des sportifs. Ici, l'utilisation de techniques d'IA devrait permettre de pallier certaines faiblesses des outils statistiques classiques dans un contexte très complexe. Gilles Bernot, Jean-Paul Comet, Guillaume Grataloup et Denis Pallez nous parlent de réseaux de régulation biologiques permettant, en particulier, de modéliser et d'analyser le comportement de certains systèmes biologiques complexes cellulaires. Il y est beaucoup question de différentes techniques d'optimisation et de leur efficacité relative, dans un secteur où la concurrence semble très active.

Dans « L'ordinateur et ses périphériques », Emmanuel Saint-James nous donne son point de vue sur un peu plus d'un demi-siècle d'évolution de l'informatique (matériel, logiciel, usage…) depuis les tubes à vide et la programmation par interrupteurs jusqu'à l'époque actuelle où presque chaque être humain a dans sa poche une puissance de calcul totalement imprévisible lorsqu'a été créé le premier microprocesseur, au début des années 70[2]. Dans cet article, il fait voyager les plus âgés d'entre nous — ceux qui ont aussi connu le plan « Informatique pour tous » — parmi tous ces grands projets, dont nombre ne furent que des miroirs aux alouettes[3].

Aujourd'hui, les deux récréations habituelles font la part belle à la littérature de science-fiction. Le défi proposé par les membres de l'équipe Codingup nous emmène dans les tréfonds d'un roman-épopée de Douglas Adams, *Le guide galactique*, qui a suscité une certaine admiration dans beaucoup de domaines… Le défi de ce mois-ci pose un petit problème d'ordonnancement tiré d'une situation particulière de ce récit : « Les cuisines Dentrassis », ou de l'art d'organiser au mieux (ici, au plus rapide) la préparation d'un repas. Pour sa part, Jean-Paul Delahaye nous propose un casse-tête avec « Les pentaminos amis » sur une façon particulière de comparer deux à deux ces espèces de pièces

2. « *I have always wished for my computer to be as easy to use as my telephone; my wish has come true because I can no longer figure out how to use my telephone* », phrase attribuée à Bjarne Stroustrup, circa 1990.

3. Comme le plan « Cinquième génération » lancé en 1982, sous l'impulsion du ministère du Commerce international et de l'Industrie japonais, et abandonné dix ans plus tard.

de puzzle — formées de la juxtaposition de cinq carrés de taille identique — en examinant certaines compositions auxquelles elles peuvent conduire. Incidemment, il paraît que lorsque l'on demande à une personne de chercher toutes les formes possibles d'un pentamino, très souvent, elle n'en trouve que 11 (au lieu des 12 réglementaires), en oubliant le pentamino le plus *simple* (comme le raconte A. C. Clarke[4]).

Et puis, avec ce numéro, nous renouons avec les fictions (la dernière était dans le numéro 21 d'avril 2021, espérons que celle-ci n'est pas simplement le chant du cygne) : Marie-Fleur nous livre une courte nouvelle, à déguster. Tout à fait d'actualité dans la problématique autour de l'IA et les polémiques qu'elle suscite, le tout traité de manière humoristique ou parfois satyrique. Nous avons là une vraie bouffée d'oxygène quand, ailleurs, certains individus — spécialistes de tout, futurologues de rien — sont capables d'écrire que «*l'intelligence devient gratuite et infiniment disponible*»…

Enfin, le bulletin (et la SIF) rend hommage à deux collègues et amis qui, chacun dans son «champ d'exercice», ont marqué notre discipline, parfois de manière très exposée, parfois dans l'ombre. Deux personnes — Jean-Pierre Archambault et Gilles Dowek — pas vraiment de la même génération, mais qui ont œuvré, parfois ensemble, pour la reconnaissance par nos tutelles de l'informatique en tant que science fondamentale. Ils ont agi continûment, avec tous les moyens dont ils disposaient, pour la promotion de cette science et pour assurer la présence de notre discipline dans les enseignements, au même titre que les autres sciences, et pas uniquement comme collection hétéroclite de technologies plus ou moins abouties.

Il ne nous reste — pour que cet hommage ne soit pas vain — qu'à poursuivre dans la voie qu'ils ont aidé à tracer.

Patrice Naudin & Denis Pallez
Comité de rédaction du bulletin *1024*
1024@societe-informatique-de-france.fr

4. Arthur C. Clarke. *Terre, planète impériale*, éditions J'ai lu, 1978, ISBN 2-277-11904-0 (version originale «Imperial Earth» publiée en 1975).

Le mot du président

1024 mérite une bonne correction !

Yves Bertrand

C'est la dernière fois que j'ai l'honneur et le plaisir de rédiger un « mot du président » pour 1024. J'appelle en effet de mes vœux qu'une successeure ou un successeur accepte de présider notre association dès 2026. Notre association est encore très jeune ; fragile ; peu connue. Elle a donc régulièrement besoin de sang neuf, d'idées neuves. Elle a également besoin de nombreux amis sûrs, dont l'amitié agit au long cours, aussi souterrainement qu'efficacement. Certains amis-partenaires sont connus, comme *Binaire*[1], l'*AEIF*[2] ou *Specif campus*[3]. Des amis-personnes, trop nombreux pour être cités, mettent leur labeur, leur notoriété, leur réseau, leurs avis au service de la SIF.

Comme ce mot ouvre 1024, je choisis — arbitrairement ! — de donner un coup de projecteur sur ceux qui œuvrent en coulisse en évoquant un travail, invisible s'il en est, qui les unit : celui de *correcteur*. Au journal *Le Monde*, une harde de correcteurs traquent dans l'ombre la moindre coquille grammaticale, orthographique ou typographique. La plupart des autres journaux ont renoncé à recourir aux compétences de correcteurs ; il y a fort à parier que les IA génératives auront raison des derniers survivants. Et nous ? Un unique correcteur agit en *ami* de tous les auteurs[4] d'articles de 1024. En ami ? Boileau[5] nous explique en quoi :

> *Un sage ami, toujours rigoureux, inflexible,*
> *Sur vos fautes jamais ne vous laisse paisible :*

1. https://blogbinaire.larecherche.fr.
2. Association des Enseignantes et Enseignants d'Informatique de France, https://aeif.fr.
3. Specif campus, https://www.specifcampus.fr.
4. **Auteur** ? **Auteure** ? **Autrice** ? **Auteurs et autrices** ? **Auteurices** ? Uniquement par souci de ne pas alourdir la rédaction, je recours ici au terme « auteur » pour la vingtaine d'occurrences de ce terme, en l'absence de terme épicène.
5. Nicolas Boileau, Art poétique, Chant I, 1674.

DOI:10.48556/SIF.1024.26.7

Il ne pardonne point les endroits négligés,
Il renvoie en leur lieu les vers mal arrangés,
Il réprime des mots l'ambitieuse emphase ;
Ici le sens le choque, et plus loin c'est la phrase.
Votre construction semble un peu s'obscurcir :
Ce terme est équivoque : il le faut éclaircir.
C'est ainsi que vous parle un ami véritable.

Pourtant, dès l'avènement de l'imprimerie, la relation entre correcteurs et auteurs est complexe, prompte à se muer en conflit larvé, parce que leurs expertises respectives s'opposent au prétexte, toujours tu, de se mesurer :

Mais souvent sur ses vers un auteur intraitable
À les protéger tous se croit intéressé,
Et d'abord prend en main le droit de l'offensé ».[6]

Recourir à un correcteur pour 1024 permet de garantir une haute qualité à l'objet langagier qu'il constitue. En effet, le correcteur agit, en cheville ouvrière, entre la fin du travail de l'auteur et le début de celui de l'éditeur. Il intervient à trois niveaux. Au premier, il rectifie une orthographe et une grammaire déficientes[7]. Facile ? Pas toujours, mais les dictionnaires permettent souvent de trancher. Le deuxième niveau concerne l'orthotypographie[8]. Quelles sont les règles en vigueur ? sont-elles uniques, incontestables ? Le troisième niveau touche à ce qu'il y a de plus sensible, de plus personnel chez un auteur : son style rédactionnel. Pour 1024, on pourrait penser que le style le plus neutre, le plus transparent par rapport au message lui-même, est de mise. Nous ne faisons œuvre ni de poésie, ni de littérature. Mais si les informaticiens que nous sommes s'intéressent d'abord aux langages formels, c'est de langue plus que de langage que nos articles sont faits. Et lorsqu'un informaticien écrit, il ne peut ni ne veut faire totalement fi de sa personnalité scripturale. Il tient à ses phrases mal fagotées, à sa ponctuation malhabile, à sa manière plus ou moins heureuse d'éviter l'écriture genrée. Et l'évolution de notre langue, transmuant en quelques décennies des termes « courants » en termes « vieillis », intégrant chaque année des centaines de mots nouveaux, ne fait rien à l'affaire. Le correcteur, bardé de sa seule connaissance et de sa seule force de conviction, ferraille à armes inégales contre un bataillon d'auteurs et contre un glissement

6. Op. cit.
7. Lorsqu'on affirme qu'auteurs et correcteurs travaillent « de conserve », commet-on un impair syntaxique ? Quel est le pluriel de « bat son plein » ? « battent son plein ? » « leur plein » ? « battent leurs pleins » ? Au fait, que signifie ici « plein » ? Quelle est sa fonction grammaticale ?
8. Quand écrit-on « professeur » avec une minuscule ? avec une majuscule ? Quelle est la taille d'une espace selon son emplacement dans la phrase ? Le manuel de l'Imprimerie nationale suffit-il à répondre à toutes ces questions ?

permanent des sens et des usages. Les auteurs ne sont pas infaillibles : répétitions, approximations ou excès de précision, références introuvables ou erronées leur sont signalées au cours du processus de transformation éditoriale. La chrysalide doit se muer en papillon pour prendre son envol auprès des lecteurs.

Alors qui l'emporte, du correcteur ou des auteurs de 1024 ? Et dans quelles conditions ? Force est de constater que nous ne l'avons jamais défini. Pour le premier niveau, il est aisé de faire vaincre le correcteur qui, retranché derrière une riche collection de dictionnaires, statue sans coup férir. Le deuxième niveau est déjà plus problématique[9] : autorise-t-on un auteur à prendre des libertés avec l'orthotypographie au seul prétexte qu'elles lui siéent ? Aucun code typographique ne fait totalement autorité. Pour le troisième niveau… j'ose un avis. Si l'auteur écrit dix fois « lycéennes et lycéens » et qu'on lui suggère que le terme épicène « élèves » allègerait sa rédaction, mais qu'il préfère sa formulation initiale, eh bien… c'est l'auteur qui a raison. Toujours. Au royaume du manuscrit, le lexicographe est roi, l'auteur est son obligé, et le correcteur est son vassal. Pour se consoler, ce dernier pourra se confire de componction au vu de l'abîme qu'il pense déceler entre ses propres compétences et celles de l'auteur. Mais… tant que la syntaxe est sauve, l'auteur a le droit d'être ampoulé, béotien, confus, digressif, évasif, futile, gouailleur, hâbleur, inepte, jobard, kafkaïen… il a le droit de maltraiter la langue tant qu'il n'en enfreint pas les règles. Le correcteur a le droit et le devoir de lui signifier explicitement sa supposée ignorance encyclopédique du « bon usage ». Et l'auteur a le droit ultime de décider s'il cède ou non aux antiennes du correcteur. En un mot comme en cent… l'auteur décide, le correcteur exécute.

Pourquoi accorder autant d'importance au fait d'administrer une « bonne correction » à 1024 ? Parce que 1024 est la vitrine concrète et persistante de notre association. Parce qu'une démarche-qualité sous-tend toutes nos actions, comme l'organisation d'une journée enseignement, de science et société ou de médiation, comme la rédaction de nos communiqués, etc. Parce que cette démarche-qualité est cruciale pour forcer le respect et l'écoute de nos nombreux interlocuteurs, notamment dans le monde socio-économique. Le rôle du correcteur de 1024 est donc central. Il est bien d'être le *rigoureux* ami de tous les auteurs. Depuis six ans, sans mot dire mais en tant de mots réécrire, ce rôle est tenu de main de maître par Patrice Naudin. La légende veut que la plupart des auteurs de 1024 murmurent : « *Ave, Patricius, morituri te salutant !* » avant même de commettre leur prose… Usant d'un insigne privilège que me confère ma fonction de président de la SIF, j'ai pu

9. Autorise-t-on des tirets d'incise demi-cadratins alors que l'usage classique impose des tirets cadratins et que l'usage moderne admet les deux (il est amusant de constater que ce point de typographie est aujourd'hui devenu un critère pour détecter la génération d'un texte par une IA) ?

ne pas lui soumettre le présent mot. Mais nul doute qu'il eut été de bien meilleure facture s'il était passé par ses fourches naudines. Salut, donc, exigeant ami des auteurs de 1024 !

Et salut, Denis Pallez, rédacteur en chef de 1024, qui a fait bien plus qu'un rédacteur en chef ne doit faire, en œuvrant en solitaire à la mise en page et à la fabrication, et qui a su judicieusement articuler son rôle avec celui de son correcteur ! Salut à deux autres amis de la SIF venus en renfort dans l'arrière-cuisine de notre bulletin : Hervé Le Crosnier et Nicolas Taffin, de C&F éditions. Ils sont certes « prestataires », mais s'investissent sans compter pour concevoir une chaîne éditoriale numérique avec 1024 et pour réaliser avec nous un travail d'orfèvre, d'artisan-éditeur, d'éditeur-militant. Ce travail est sans doute peu visible à l'œil nu, sauf pour les amoureux des polices de caractères, de l'orthotypographie et de l'édition en général, qui savent que le niveau de professionnalisme atteint avec ce n° 26 de 1024 est peu commun. Salut à Bruno Mermet, imminent successeur de Denis. Il pourra tenir son rôle de « rédac-chef », libéré de la technologie éditoriale grâce à la professionnalisation permise par Denis. Salut aux amis de 1024, salut à tous les amis de la SIF !

Quelques difficultés de l'enseignement de l'algorithmique et de la programmation aux débutants

Pierre Tchounikine
Université Grenoble Alpes

Les travaux de recherche sur l'enseignement de l'algorithmique et de la programmation aux débutants confirment un certain nombre de constats que les enseignants connaissent bien : les élèves et les étudiants ne développent souvent qu'une compréhension rudimentaire des notions enseignées ; les échecs et abandons sont nombreux ; les résultats sont souvent polarisés avec un groupe qui réussit bien et un autre en grande difficulté.

Cet article propose un éclairage sur trois difficultés spécifiques, différentes mais non indépendantes, qui contribuent à cette situation : le nombre de notions abordées et leurs interrelations ; le rôle des modèles mentaux ; et la question du transfert. Ces problèmes se posent tant au niveau des L1 universitaires que de l'enseignement au lycée (j'utiliserai le mot « étudiant » comme un terme générique pour « étudiant », « étudiante » et « élève »). Les éléments de réflexion proposés ici s'appuient sur les travaux de recherche sur l'enseignement de l'informatique et notamment les synthèses présentées dans [2,5,6], mes travaux de recherche personnels et une longue pratique de l'enseignement en L1.

DOI:10.48556/SIF.1024.26.11

Densité et complexité de l'articulation notionnelle

Pour expliquer ce qu'est un programme informatique il faut en montrer un. Le respect des traditions peut amener à commencer par le traditionnel «*Hello World*» (en C, figure 1.a). Le souhait d'en profiter pour présenter aux étudiants un programme plus proche des exercices à venir peut amener à utiliser un exemple comme celui proposé en figure 1.b (en Python).

```c
#include <stdio.h>
int main(void)
{
printf("Hello World! \n");
return 0;
}
```

(a)

```python
def pg1():
  print("Merci de donner votre nom")
  nom=input()
  print("Bonjour", nom)
  rep=input("Vous avez déjà fait de l'informatique oui/non ?")
  if rep=="oui":
    print("Bon, c'est bien, mais attention ...")
  else:
    print("Pas de souci, on part de zéro, mais ...")
  print("Il va falloir bosser !!!")
```

(b)

Fig. 1. Exemples de programmes présentables en début d'enseignement.

Prenons le programme Python. Lorsqu'on le présente à des étudiants débutants ils devinent sans difficultés comment il fonctionne et ne sont pas surpris par ce que produit son exécution (sauf parfois par le fait que le dernier «`print`» s'exécute dans tous les cas).

Relisons maintenant le code en nous demandant : quelles sont les notions nécessaires à la compréhension de ce programme et de son exécution? Il est possible de prendre cette question à différents niveaux de granularité, mais ce qui est sûr c'est que la liste des notions impliquées est longue : programme et fonction; langage, syntaxe, sémantique, interpréteur, exécuteur; mots clés, syntaxe graphique, indentation; instruction (simple, complexe); variable, identificateur, place mémoire, affectation, type, chaîne de caractère, booléens (ainsi que les entiers ou les réels si l'on prend un exemple numérique); expression, égalité; conditionnelle; entrée-sorties, fonctions «`print`» et «`input`» (et donc, possiblement, paramètre, effet de bord, librairie?); et la liste n'est pas exhaustive.

Ce que cet exemple illustre c'est qu'un programme Python basique — du type de ceux dont on pense qu'il est légitime d'attendre des étudiants qu'ils soient capables de le construire eux-mêmes dès les premières semaines d'enseignement — implique un nombre très important de notions. En fait, ce programme Python mobilise une bonne partie des notions abordées dans un cours d'initiation.

Premier point important : cette complexité n'est pas liée à l'exemple choisi, elle est inhérente à l'informatique. Le programme «*Hello World*» implique

également un bon nombre de notions, et les explications que l'on doit fournir aux étudiants pour qu'ils comprennent une instruction comme «`x = 1`» impliquent bien plus que les notions de variable et d'affectation. Il n'est bien évidemment pas nécessaire que les étudiants maîtrisent parfaitement toutes les notions listées ci-dessus pour commencer à programmer mais, que l'on commence l'enseignement en présentant du code ou pas, il est difficile de ne pas les aborder, d'une façon ou d'une autre, dès les premières séances.

Second point important : la plupart des notions impliquées sont fortement interreliées. Quelle que soit la façon dont on présente les choses, l'étudiant débutant doit donc percevoir et articuler de façon concomitante (plutôt que : une à une, et petit à petit) des notions qui, par ailleurs, relèvent de différents registres. Ainsi, les notions de «variable», «mémoire», «type», «affectation» et «évaluation» sont de natures très différentes, mais il est difficile de les aborder indépendamment les unes des autres.

L'une des caractéristiques importantes de l'enseignement de l'informatique à des débutants est donc la densité et la complexité de l'articulation notionnelle. En quelques séances, les étudiants doivent comprendre ou, *a minima*, développer une perception à peu près cohérente de ce qui, représenté sous forme d'une carte conceptuelle, correspond à un graphe de plusieurs dizaines de nœuds densément interconnectés. Le nombre de notions à présenter avant de pouvoir commencer à faire des exercices impose aux enseignants d'adopter un rythme soutenu, et les étudiants débutants se voient quasi-instantanément plongés dans un nouveau monde conceptuel auquel ils doivent donner sens très vite.

Les travaux de recherche montrent que cette caractéristique joue un rôle important dans les difficultés des étudiants et la polarisation des résultats. Nous (humains) apprenons de façon constructive, en accrochant de nouvelles connaissances à celles dont nous disposons déjà et en bousculant, étendant ou corrigeant celles-ci. Assimiler en même temps plusieurs notions, en particulier lorsqu'il est difficile de les percevoir à l'aune de ce que nous connaissons déjà (et qu'il y a de plus des faux amis, la notion de variable, par exemple) est intrinsèquement difficile et déstabilisant.

Les premières séances d'informatique sont donc très différentes de celles d'autres domaines — maths, physique, etc. — dans lesquels les étudiants ont déjà une base notionnelle sur laquelle s'appuyer pour comprendre et assimiler ce qui est vu en cours (domaines où, par ailleurs, les premiers cours reprennent souvent des éléments de l'année précédente, ce qui peut encore accroître la déstabilisation des étudiants face aux premiers cours d'informatique où, dès le départ, les étudiants doivent comprendre de nouvelles notions). Lorsque chaque cours introduit plusieurs notions nouvelles et que leur maîtrise est nécessaire à la compréhension de celles du cours suivant, la moindre difficulté contamine le reste. Par ailleurs, le fait d'apprendre et

réussir crée de la motivation et de la confiance, ce qui facilite les apprentissages suivants, et à l'inverse ne rien comprendre de ce qu'il se passe crée une dynamique négative, surtout quand les autres étudiants ont l'air de trouver cela facile et que les messages d'erreurs et autres bugs amènent à développer des frustrations et des émotions négatives. Il s'enclenche donc des dynamiques positives (si l'on comprend assez rapidement les principes du programme Python de la figure 1, la suite — boucles, conditionnelles imbriquées, etc. — ne pose généralement pas de difficulté spécifique) et négatives (si l'on ne comprend pas très vite, la suite devient impossible). Dit autrement : si le fameux « déclic » (que je reformulerais en « l'étudiant dispose d'une carte conceptuelle suffisamment complète et cohérente pour lui permettre de comprendre à peu près ce qu'il se passe et commencer à réussir quelques tâches ») ne se fait pas rapidement, les étudiants présentant les atouts généraux de la réussite (assidus, motivés, tenaces, sans soucis matériels ou personnels trop importants) peuvent rattraper le coup, mais pour les autres cela devient très vite mission impossible.

Les implications pour la conduite de l'enseignement sont assez directes :

– expliquer aux étudiants cette caractéristique de l'enseignement de l'informatique pour les aider à ne pas paniquer et à comprendre les efforts qu'ils doivent faire (« il faut travailler dès le début » est une consigne générale qui vaut pour tous les enseignements, mais qui prend un sens et une importance spécifiques en informatique) ;
– s'assurer que les étudiants comprennent les notions et ne développent pas des modèles mentaux obérant la suite des enseignements (cf. section suivante) en leur demandant de reformuler les choses avec leurs mots (par exemple, ce qu'est un type ou une structure de contrôle) ou encore de critiquer des bouts de code qui « marchent » mais sont incorrects ou discutables (par exemple des constructions comme « *expression booléenne* `== true` » ou l'utilisation d'un « `return` » qui brise indûment une conditionnelle ou une boucle) : contrairement à beaucoup d'autres domaines les étudiants peuvent utiliser des notions d'une façon qui leur semble satisfaisante ou, en tout cas, leur permet d'obtenir un programme « qui marche », alors qu'en fait ils ne les comprennent pas vraiment ;
– ne pas penser que s'il y a un groupe d'étudiants qui avance bien cela veut dire que les autres ne travaillent pas assez (de façon contre-intuitive, les étudiants ayant rapidement compris les bases peuvent avancer et réussir en travaillant beaucoup moins que certains autres qui se débattent et s'enferrent en essayant de suivre sans maîtriser les concepts de base) ;
– réfléchir avec attention aux choix pédagogiques que l'on opère (choix des notions traitées et des notions omises ou seulement mentionnées, degré de simplification — voire, de correction — auquel on présente les choses) en fonction des étudiants et des objectifs à terme de l'enseignement (par

exemple, insister dès le départ sur la notion de type pour préparer le passage du paradigme impératif au paradigme orienté objet) ;
– garder en tête que des explications qui sont claires et pertinentes lorsque l'on dispose d'une carte conceptuelle cohérente (comme c'est le cas pour l'enseignant) peuvent être incompréhensibles, voire contre-productives, lorsque la carte conceptuelle avec laquelle on les reçoit est clairsemée ou inexacte (comme c'est le cas, à des degrés variables, pour tous les étudiants).

Il est également possible de chercher à gérer ces difficultés en réfléchissant à l'organisation des enseignements et à sa cohérence avec les points évoqués ci-dessus. En effet, la structure d'enseignement standard (enseignement à un groupe d'étudiants, TP et projets en binômes) amène à gérer de façon synchrone des étudiants qui ne peuvent pas avancer au même rythme, et il est difficile d'aider les étudiants qui présentent des difficultés conceptuelles sans abandonner ceux qui ont passé l'obstacle et dont il faut nourrir la motivation.

Pour essayer de gérer ou au moins de limiter cet écueil il est possible de s'inspirer des initiatives visant à l'aborder de façon radicale, par exemple les modèles de type « *mastery learning* » : le programme d'enseignement est découpé en étapes ciblant chacune un petit nombre de notions et de compétences ; pour chaque étape, l'étudiant dispose de ressources lui permettant de travailler à son rythme (par exemple, documents ou vidéos et exercices autocorrigés) ; lorsque l'étudiant se sent prêt il passe un test (par exemple, des exercices de programmation tirés au hasard d'une banque d'exercices) ; si le test démontre une bonne maîtrise des notions et compétences de cette étape il passe à la suivante, et sinon il continue à travailler et s'entraîner à l'étape courante avant de repasser le test (jusqu'au succès, sans pénalité liée au nombre d'essais). Différentes formes de support additionnel peuvent être proposées, par exemple le fait de passer les tests avec des enseignants ou tuteurs (ce qui permet de faire un point avec l'étudiant) ou encore de proposer des séances de questions-réponses apportant à la fois une aide et un rythme de référence. Confer, par exemple, les expériences relatées dans [3,4].

La mise en œuvre de ce type de modèle pose des problèmes organisationnels importants (et probablement rédhibitoires dans la plupart de nos structures d'enseignement), et par ailleurs n'a pas que des avantages. Ainsi, si ce type d'enseignement semble amener les étudiants à développer des connaissances moins parcellaires et moins fragiles, il ouvre la porte à des comportements de procrastination ou d'efforts *a minima* (juste suffisants pour passer le test)[1]. Les principes, propriétés et limites de cette approche

1. Il y a là un point de discussion incident : que faire d'une méthode d'enseignement qui permet aux étudiants qui jouent le jeu de mieux réussir mais qui donne de moins bons résultats que la méthode d'enseignement standard pour les autres, surtout quand ces derniers sont les plus nombreux ?

peuvent cependant être une source d'inspiration pour l'organisation des enseignements (rythme, contrôle continu, etc.) et l'attention aux étudiants au sein de nos structures plus traditionnelles.

Autre source d'inspiration : les travaux de recherche sur la charge cognitive. Les situations de résolution de problème amènent les étudiants à engager des efforts cognitifs liés à la tâche (au fait de comprendre et résoudre le problème) et à l'apprentissage (au fait d'apprendre). Les travaux montrent que, lorsque les efforts liés à la tâche sont trop importants, cela affecte négativement les apprentissages. Ce constat général s'applique directement à nos exercices d'informatique standards (« construire un algorithme ou un programme qui… »), avec le facteur aggravant que la complexité des interrelations entre les notions impliquées augmente fortement la charge cognitive. Faire travailler les élèves sur des exercices corrigés (lecture et analyse de programmes existants) ou sur des exercices d'arrangement et de modification de bouts de code sont des moyens de limiter cet écueil. Il est bien évident que l'on ne devient pas compétent en algorithmique et en programmation sans construire des algorithmes et des programmes : les activités de résolution de problème sont fondamentales. Cela ne veut pas dire qu'il ne faut enseigner que via ce type d'activité.

Rôle des modèles mentaux

Pour réfléchir et résoudre des problèmes nous (humains) avons besoin d'outils psychologiques qui nous aident à structurer nos processus de pensée. Nous développons donc des modèles mentaux de, par exemple, la notion de variable ou de ce qu'il se passe quand on lance l'interpréteur Python sur le code présenté en figure 1. Sans surprise, les modèles mentaux que développent les étudiants ne sont pas toujours corrects ni pertinents, et cela crée des obstacles aux apprentissages.

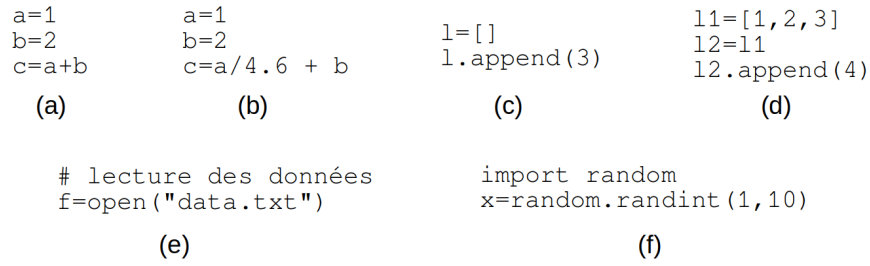


Fig. 2. Affectations en Python.

Prenons les instructions Python présentées en figure 2. Le modèle selon lequel une variable est « une boîte avec une valeur dedans »[2] et une affectation est « le fait de mettre une valeur dans la boîte », modèles que l'on enseigne ou que les étudiants construisent spontanément, permettent de comprendre certains aspects de ce qu'il se passe quand on lance l'interpréteur, mais certains seulement : cela marche bien pour le code (a) ; cela ne rend pas compte du transtypage en (b) ou encore de la première instruction en (c), qui vise moins à « mettre une liste vide dans l » qu'à indiquer que l est une liste et donc permettre les opérations associées aux listes ; cela marche assez bien pour expliquer et comprendre une partie de l'accès partagé créé en (d), mais une partie seulement ; pour (e), ces modèles marchent mal. Les modèles véhiculant l'idée qu'une variable c'est « un identifiant, un type et une valeur », et qu'un type c'est « un ensemble de valeurs et un ensemble d'opérations » permettent de compléter la vision de ce qu'il se passe pour certains de ces cas (mais il y a peu de chances que les étudiants développent spontanément ce type de modèle, d'où l'importance des points d'attention mentionnés précédemment).

De même, les modèles selon lesquels « un ordinateur c'est essentiellement un processeur qui travaille sur de la mémoire » et « chaque ligne d'un programme Python définit une action qui va être appliquée » marchent assez bien. Un étudiant ayant développé cette façon de voir les choses a peu de raisons de la remettre en question : il n'y a pas besoin de déclarer les variables en Python, les instructions que les étudiants sont amenés à lire ou écrire sont très souvent des actions sur des variables, et pour les lignes de type « `if` » ou « `while` » la notion d'action peut être facilement étendue à « passer à un autre bout de code ». Malheureusement, cela va poser des soucis pour comprendre les typages évoqués ci-dessus (une ligne comportant une affectation a en fait plusieurs effets) ou les codes en (e) et (f). Comprendre que, outre la spécification d'une action sur des variables, une ligne de code puisse avoir des finalités aussi différentes que de définir une structure (e.g. une fonction), d'aider les programmeurs qui reliront le programme (ou nourrir le générateur de documentation), permettre la gestion de la mémoire et la vérification de la cohérence des opérations (typage), donner accès au contenu de librairies ou interfacer des opérations liées au système d'exploitation est loin d'être intuitif. Et, bien entendu, expliquer tout cela dès que l'on commence à lire ou écrire des programmes, i.e. dès les premiers cours, n'est probablement pas une bonne idée (cf. section précédente).

Première implication : il est important d'enseigner, très explicitement, des modèles permettant de comprendre ce qu'il se passe à l'exécution. Ne pas le faire (ce qui, d'après les enquêtes, est assez fréquent), c'est-à-dire laisser les

2. La façon de parler des notions informatiques que je reprends ici correspond au type de discours que l'on entend fréquemment dans un cours d'initiation, pas à ce que je propose d'enseigner.

élèves développer un modèle mental idiosyncratique de comment fonctionne l'exécution du code, ouvre la porte à des conceptualisations potentiellement contre-productives. Il est d'autant plus important d'enseigner ces modèles que lorsque nous (humains) avons développé un modèle qui nous est utile, il nous est très difficile d'en changer, et c'est un comportement assez rationnel. Il ne suffit donc pas d'expliquer comment il faut voir les choses, i.e. le modèle dont on pense qu'il va le mieux aider les étudiants à un stade donné d'apprentissage, il faut également identifier et déstabiliser les façons de voir qui sont improductives ou délétères.

Seconde implication : un modèle étant une simplification de la réalité guidée par un but, le but poursuivi et l'intention du modèle (ce qu'il permet de comprendre et de faire) sont des choix d'enseignement majeurs. Comme on l'a vu au début de cet article, l'un des besoins les plus urgents et importants de la plupart des étudiants est de gérer la masse de nouvelles notions (et d'interrelations entre ces notions) des premiers cours. Les modèles proposés aux étudiants devraient donc, *a minima*, prendre en compte ce point.

Ainsi, et pour prendre un exemple non anodin et non consensuel de lien entre les deux écueils évoqués ci-dessus (complexité notionnelle, modèles mentaux), l'initiation à la programmation en Python commence souvent par taper quelques instructions dans la console et regarder ce que cela donne ; l'étape suivante consiste à créer un fichier `.py` dans lequel on écrit un ensemble de lignes effectuant une tâche (e.g. lire les coefficients d'une équation de second degré puis afficher les solutions) et à évaluer le contenu du fichier ; quelques semaines plus tard, on structurera le code en fonctions. La progression semble logique : on part du plus simple et on complexifie. Ceci étant, on prend le risque que les étudiants développent des modèles mentaux inadéquats de ce qu'est un programme ou une fonction (notamment une confusion entre un programme et un fichier contenant des lignes de code), et que cela leur rende plus difficile le passage vers la définition de fonctions ou encore la compréhension des problèmes que posent les variables globales. Dans la mesure où, en toute hypothèse, la notion de programme est (explicitement ou implicitement) présente dès le premier cours, chercher à baisser la complexité notionnelle en écrivant des lignes de code sans les structurer en fonctions ne présente pas que des avantages[3].

3. Autre approche possible : le premier extrait de code montré est une fonction (figure 1.b) et, dès les premiers TD et TP, les étudiants définissent des fonctions et les exécutent en invoquant leur nom (comme ils le font pour lancer leurs applications sur leur smartphone ou leur jeux sur ordinateur). La distinction entre fichier et programme est claire (un fichier peut contenir une ou plusieurs fonctions), et par la suite les notions de procédure et de fonction (au sens de sous-programmes) et de paramètres apparaissent comme un affinement notionnel (et non comme un bouleversement d'une façon de procéder que l'on avait eu du mal à apprendre et « qui marche »).

L'une des spécificités de l'informatique est que les rétroactions de la machine (le compilateur ou l'interpréteur signale les erreurs de syntaxe, l'exécution du programme permet de constater s'il fait ce que l'on avait prévu ou pas) permettent d'aborder certaines parties de l'enseignement de façon « descendante » (typiquement : expliquer une notion puis l'utiliser) mais également, dans une certaine mesure, « ascendante » (typiquement : proposer aux élèves de tester des choses sur machine, puis donner du sens à ce qu'ils ont constaté). Comme les étudiants peuvent créer des programmes sans vraiment comprendre toutes les notions impliquées, ce peut être (et, en tout cas, c'est parfois utilisé comme) une façon de résoudre le problème du nombre de notions à enseigner. Que la pratique soit utilisée comme une application d'un enseignement préalable ou pour amener les étudiants à commencer à comprendre par eux-mêmes via une succession d'essais et erreurs, la centralité des activités de programmation et du couperet de l'exécution par la machine (« ça marche » ou « ça ne marche pas ») a un effet majeur sur les modèles mentaux que développent les étudiants. Comme le fait qu'un programme « marche » ou « ne marche pas » peut être lié à différentes choses, et n'est en tout état de cause pas le point le plus important, vérifier (dès les premiers cours) que les étudiants développent des conceptualisations cohérentes est un enjeu majeur.

Problème du transfert

Le transfert se définit habituellement comme le mécanisme par lequel nous réutilisons nos connaissances dans un nouveau contexte.

En informatique, les phénomènes de transfert vont par exemple jouer un rôle dans la façon dont les étudiants vont aborder leur second langage de programmation. Ainsi, les connaissances acquises dans le contexte de la programmation par agencement de blocs (e.g. en Scratch) peuvent faciliter ou, au contraire, rendre plus difficile, le passage à un langage textuel comme Python. Les conceptions développées via l'usage d'un langage de programmation impératif vont influer sur le passage aux langages fonctionnels ou orientés objets. Etc.

Pour les débutants, le point sur lequel la question du transfert joue un rôle central et majeur relève cependant de l'algorithmique : il s'agit du transfert entre situations de résolutions de problèmes, i.e. de la capacité à se rendre compte que le problème à résoudre présente des similarités avec des problèmes déjà rencontrés, et de s'en servir pour résoudre ce nouveau problème. Pour prendre quelques exemples typiques : se rendre compte que « calculer la moyenne d'une liste de températures » c'est la même chose que le « calculer la moyenne d'une liste de notes » que l'on a traité en cours, et que l'on peut donc utiliser le même schéma algorithmique ; que « chercher le

maximum d'une liste de températures » présente des similarités avec « calculer la moyenne d'une liste de températures » (il faut passer sur tous les éléments, on peut donc utiliser le même type de boucle), mais que le traitement à effectuer est différent ; ou encore que « déterminer s'il y a un 20 dans la liste des notes » nécessite, comme pour « calculer la moyenne d'une liste de notes », de parcourir cette liste (il y a des éléments communs réutilisables donc), mais que l'on n'a pas toujours besoin de parcourir toute la liste et qu'il faut donc faire attention aux conditions d'arrêt.

En tant qu'enseignants nous sommes souvent perplexes, pour ne pas dire désemparés, devant l'incapacité de certains étudiants à voir que le problème proposé est similaire, voire identique à l'habillage près, à un exercice déjà traité. Pourtant, cela correspond tout à fait aux résultats des travaux de recherche sur le transfert. Il a été montré que les humains ont des capacités générales de transfert (mais le fait que l'on puisse les améliorer par apprentissage est une question ouverte). En tant qu'enseignants, nous nous appuyons plus ou moins implicitement sur ces capacités. Cependant, les travaux empiriques montrent que cela marche souvent assez mal, et que les résultats sont généralement très en-deçà de nos espérances (et de nos croyances). Si l'on recentre sur l'informatique, et pour reprendre une polémique ancienne mais toujours très actuelle : l'hypothèse selon laquelle l'algorithmique et la programmation améliorent les capacités de résolution de problème était au cœur des travaux de Seymour Papert sur Logo (dont Scratch est le successeur), les travaux empiriques des années 80 ont mené à la conclusion que ce n'était généralement pas le cas, mais cette hypothèse (positive, sympathique, enthousiasmante, gratifiante) reste cependant solidement ancrée dans la tête de la plupart des enseignants d'informatique et, par exemple, du renouveau des idées de Papert via la notion de « pensée informatique » (*computational thinking*) proposée par Wing [7]. S'il y a certes des expériences positives, la plupart des travaux sur la question montrent malheureusement des résultats modestes, voire inexistants [1]. (Ceci ne doit cependant pas nous conduire à arrêter les travaux sur le sujet, notamment car la question du transfert pose des problèmes méthodologiques qui rendent les travaux extrêmement complexes, ce qui peut et doit nous amener à considérer les résultats actuels, tant positifs que négatifs, avec précaution.)

L'une des raisons qui rendent ce constat (et donc sa prise en compte) difficile est que les mécanismes d'abstraction et d'instanciation et l'utilisation de schémas (au sens large : types de boucle, schémas algorithmiques, design patterns, types de problèmes, frameworks) pour analyser les situations et résoudre les problèmes sont des compétences fondamentales de l'informatique. En tant qu'experts nous « voyons » les similarités (c'est le propre de l'expertise) et nous savons que cette compétence est centrale. Mais, par

définition, les débutants ne sont pas des experts, et il est toujours difficile de savoir ce que « voient » les autres.

Les implications sont assez simples : il ne faut pas penser que multiplier les exemples d'un schéma algorithmique suffit pour que les étudiants en infèrent des structures abstraites pertinentes qu'ils pourront mobiliser pour interpréter et résoudre de nouveaux problèmes similaires ; le transfert spontané marchant très mal, il faut aider les étudiants à détecter les similarités entre problèmes et entre solutions.

La difficulté qui se pose alors est que le fait d'enseigner d'une façon qui favorise le transfert peut entrer en tension avec d'autres considérations. Ainsi, l'enseignement à un niveau abstrait (par exemple, l'enseignement de schémas algorithmiques génériques) favorise le transfert mais, malheureusement, les travaux montrent qu'un enseignement à un niveau abstrait est souvent très difficile pour des débutants (ils n'ont pas assez de connaissances pour relier ces abstractions à des choses qu'ils connaissent déjà et les comprendre vraiment). Inversement, proposer des exercices instanciés sur des cas concrets et des domaines sémantiques familiers des étudiants (par exemple, la gestion de notes) facilite l'apprentissage mais, malheureusement, les détails de surface ont souvent un effet négatif sur les modèles et les schémas de résolution que développent les étudiants. De même, faire travailler les étudiants débutants sur des exercices résolus est très efficace (beaucoup plus que de les laisser sécher sur des tâches de construction d'algorithmes ou de programmes) mais, malheureusement, amène à travailler sur des cas particuliers et ne favorise donc pas l'abstraction.

Il faut donc trouver le difficile et délicat équilibre permettant d'aider les étudiants à aller au-delà des exercices travaillés sans les perdre par trop d'abstraction. Typiquement, proposer à des étudiants débutants d'analyser un exercice comme un problème d'optimisation est sans doute un peu ambitieux. En revanche, analyser une répétition en termes de types de boucle (« Pour », « Tant que ») est devenu une pratique standard (dit autrement : ces structures se sont révélées des abstractions accessibles et utiles). Mettre en évidence des schémas algorithmiques de type « boucle de lecture et vérification de données » (lire puis relire une donnée en boucle tant qu'elle ne respecte pas les critères de validité) ou « application d'un traitement à l'ensemble des éléments d'une liste » est une option possible. Autres exemples : « boucle pour chercher combien » (et la notion d'accumulateur), « boucle pour chercher si » (et les notions de critères et de drapeau), « boucle pour chercher tous les », etc. Quels que soient les choix opérés, les travaux montrent que l'introduction de termes permettant de nommer des buts intermédiaires (ce qu'il faut faire pour mettre en place les schémas algorithmiques enseignés) contribue aux transferts futurs.

Références

[1] Peter J. Denning, and Matti Tedre. 2019. *Computational thinking*. MIT Press.

[2] Mark Guzdial, and Benedict du Boulay. 2019. The History of Computing Education Research. In S. A. Fincher & A. V. Robins (Eds.) *The Cambridge Handbook of Computing Education Research*. Cambridge, UK: Cambridge University Press, p. 11–39.

[3] Brendan McCane, Claudia Ott, Nick Meek, and Anthony V. Robins. 2017. Mastery learning in introductory programming. In *Proceedings of the Nineteenth Australasian Computing Education Conference*, ACM, New York, NY, USA, p. 1–10.

[4] Claudia Ott, Brendan McCane, and Nick Meek. 2021. Mastery Learning in CS1-An Invitation to Procrastinate?: Reflecting on Six Years of Mastery Learning. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V1*, New York, p. 18–24.

[5] Anthony V. Robins. 2019. Novice programmers and introductory programming. In S. A. Fincher & A. V. Robins (Eds.) *The Cambridge Handbook of Computing Education Research*. Cambridge, UK: Cambridge University Press, p. 327–376.

[6] Anthony V. Robins, Lauren E. Margulieux, and Briana B. Morrison. 2019. Cognitive sciences for computing education. In S. A. Fincher & A. V. Robins (Eds.) *The Cambridge Handbook of Computing Education Research*. Cambridge, UK: Cambridge University Press, p. 231–275.

[7] Janet Wing. 2006. Computational thinking. *Communications of the ACM*, 49(3), p. 33–35.

Ressources logicielles libres et open source dans l'enseignement de l'informatique en MP2I et MPI

Aslı Grimaud
Lycée Guy Mollet, Arras

L'enseignement de l'informatique en CPGE MP2I et MPI repose sur une formation exigeante, à la fois théorique et pratique, visant à transmettre une compréhension rigoureuse des fondements scientifiques de l'informatique et de ses implications techniques et sociétales. Dans ce contexte, l'usage de logiciels libres et *open source* constitue un levier pédagogique essentiel. Cet article présente trois outils emblématiques de cette approche : le système d'exploitation GNU/Linux, l'environnement de gestion de versions Git accompagné de GitLab, et l'éditeur de texte collaboratif HedgeDoc. En plus de leurs qualités techniques, ces outils incarnent une philosophie fondée sur l'ouverture, la transparence, le partage des connaissances et la maîtrise de l'environnement numérique. En plaçant l'informatique libre au cœur de la formation, on offre aux étudiantes et aux étudiants un environnement cohérent et éthique.

Les filières MP2I (mathématiques, physique, informatique et ingénierie) et MPI (mathématiques, physique, informatique), introduites en 2021 dans les classes préparatoires aux grandes écoles (CPGE), constituent une avancée majeure dans l'intégration de l'enseignement de l'informatique au sein des formations scientifiques postbac. Elles répondent à un besoin croissant de former des étudiants à la science informatique, non seulement comme un ensemble de compétences techniques, mais aussi comme une discipline scientifique à part entière. L'objectif de cet enseignement ne se limite pas à

DOI:10.48556/SIF.1024.26.23

l'acquisition de compétences techniques : il vise à former des personnes informées, capables de comprendre et de maîtriser les techniques numériques tout en mesurant les enjeux. Les filières MP2I et MPI préparent ainsi les étudiants à devenir les ingénieurs, les enseignants, les chercheurs de demain, mais aussi, plus largement, des citoyens capables de gouverner leur vie professionnelle et personnelle en s'appuyant sur une démarche scientifique rigoureuse et une conscience éclairée des transformations induites par l'informatique dans la société contemporaine [4].

Dans ce contexte, l'utilisation de ressources logicielles libres et *open source*[1] prend une importance stratégique. Un logiciel est dit « libre » s'il garantit à ses utilisateurs un certain nombre de libertés : liberté d'utiliser, d'étudier, de modifier et de redistribuer. Ces libertés s'accompagnent souvent d'une transparence du code source et d'une dynamique communautaire forte. Ce modèle s'oppose au logiciel propriétaire, dont le fonctionnement est opaque, figé, et généralement soumis à des conditions de licence restrictives. Les logiciels *open source*, tout en suivant largement les principes du logiciel libre, insistent davantage sur l'accès au code source et la collaboration technique ouverte.

Dans l'enseignement supérieur, les outils libres sont utilisés de longue date pour l'enseignement de la science informatique. Les CPGE, toutefois, relèvent administrativement du secondaire et héritent de l'infrastructure informatique des lycées, souvent conçue pour d'autres usages pédagogiques et dépourvue d'équipe technique dédiée à la gestion de systèmes complexes. Cette configuration rend l'adoption de solutions logicielles libres plus difficile, en particulier dans les filières MP2I et MPI, où l'informatique est l'une des disciplines principales. La mise en place d'un environnement cohérent et libre repose donc encore largement sur des initiatives locales, gérées au cas par cas.

L'utilisation de logiciels libres et *open source* en classe, notamment en informatique, présente plusieurs avantages fondamentaux :

– sur le plan pédagogique, ces logiciels encouragent une posture active et exploratoire. Les étudiants ne se contentent pas d'utiliser un logiciel : ils en découvrent le fonctionnement interne, les principes de conception, et la manière dont des projets logiciels réels sont pensés, développés, documentés et maintenus. Cette immersion progressive dans l'écosystème du libre favorise une compréhension de la logique collaborative, de la traçabilité des modifications, et de la transparence, autant d'éléments structurants pour une culture informatique solide ;
– sur le plan technique, ces outils permettent aux étudiants d'explorer concrètement les mécanismes internes des logiciels, bien au-delà d'une

1. Selon les expressions consacrées « *Free and open-source software (F/OSS, FOSS)* » ou « *free/libre/open-source software (FLOSS)* ».

simple utilisation en surface. Ils peuvent commencer par personnaliser des éléments, comprendre l'architecture des fichiers, manipuler des configurations, et ainsi mieux appréhender la structure d'un système informatique. Cette approche favorise également la compréhension de la gestion des fichiers, des permissions, et plus largement de l'organisation d'un ordinateur, souvent peu visible dans les environnements propriétaires. Elle ouvre aussi la voie à des notions clés telles que la sécurité informatique ;

– sur le plan éthique, les logiciels libres incarnent des valeurs essentielles pour toute formation à l'informatique : liberté d'usage, partage des connaissances, autonomie technologique et responsabilité individuelle et collective. Ils permettent d'initier les étudiants à des réflexions sur la souveraineté numérique, l'accessibilité du savoir, ou encore le rapport entre technologie et pouvoir. En découvrant qu'un logiciel peut être conçu, amélioré, partagé et utilisé librement par une communauté, les étudiants sont sensibilisés à une vision ouverte et émancipatrice de la technologie, en opposition à une logique de consommation fermée.

Cette orientation est d'ailleurs encouragée par les instances officielles. Le rapport n° 22-23-006A de l'IGÉSR [3], «*La préparation aux formations et aux métiers du numérique et de l'informatique…* », recommande explicitement de prendre en compte les besoins pédagogiques spécifiques à l'enseignement de l'informatique dans les politiques d'équipement des établissements (recommandation, section 2.4.4). Il souligne notamment que ces besoins incluent l'accès à des environnements logiciels adaptés, tels que les systèmes d'exploitation libres.

Dans cet article, on présente trois exemples concrets d'utilisation de ressources libres et *open source* en CPGE MP2I et MPI : le système d'exploitation GNU/Linux, l'environnement de gestion de version Git (et GitLab), et enfin l'éditeur collaboratif HedgeDoc, utilisé pour la prise de notes et la collaboration en classe. Ces outils, au-delà de leur efficacité technique, s'inscrivent dans une démarche pédagogique cohérente avec les objectifs de ces filières.

La mise en place d'un environnement informatique libre et cohérent au sein d'un établissement scolaire ne va cependant pas sans difficultés. Installer, configurer et maintenir des outils libres tels qu'un système GNU/Linux, un service de gestion de versions ou un éditeur collaboratif nécessite des compétences techniques avancées, que ce soit du côté des enseignants ou du côté des personnels techniques. La gestion d'une salle informatique avec un réseau local dédié, une connectivité stable, et parfois même des serveurs internes (physiques ou virtuels) adaptés aux besoins pédagogiques, requiert un investissement conséquent en temps et en savoir-faire. Il s'agit d'un véritable travail d'administration système et réseau, bien souvent

assumé de manière artisanale par des enseignants passionnés mais insuffisamment accompagnés.

À cela s'ajoutent des contraintes institutionnelles et techniques, telles que les restrictions imposées par certains pare-feux ou politiques de sécurité réseau, qui peuvent limiter l'accès aux ressources nécessaires pour l'enseignement de l'informatique. Ces freins techniques, lorsqu'ils ne sont pas anticipés ou levés à temps, ralentissent la mise en œuvre pédagogique des outils libres et réduisent l'ambition initiale des programmes. Dans ce contexte, l'utilisation de ressources en ligne reposant sur des serveurs externes, parfois opaques sur le traitement des données, tend à se généraliser, souvent par facilité, au détriment d'un enseignement maîtrisé, et conforme aux exigences pédagogiques de la discipline. Ce constat appelle à une reconnaissance structurelle forte de ces besoins spécifiques, et à une meilleure articulation entre les exigences de sécurité informatique et les impératifs pédagogiques, afin de garantir des conditions d'enseignement à la hauteur des enjeux de la formation en informatique.

Système d'exploitation GNU/Linux : outil d'apprentissage au service de la compréhension informatique

Le système GNU/Linux est emblématique de la philosophie du logiciel libre. Le projet GNU, initié par Richard Stallman en 1983 [5], visait à créer un système d'exploitation entièrement libre, composé d'outils, de bibliothèques et de programmes conformes aux principes de liberté logicielle. Il manquait toutefois un noyau fonctionnel pour que le système soit complet. C'est en 1991 que Linus Torvalds publie la première version du noyau Linux, qu'il place rapidement sous une licence libre [6]. La combinaison du système GNU et du noyau Linux a donné naissance à ce qu'on appelle aujourd'hui GNU/Linux, utilisé dans une multitude de contextes : serveurs, supercalculateurs, smartphones (*via* Android), objets connectés, etc.

Installation native

Utiliser un système GNU/Linux installé en natif constitue la solution la plus cohérente et la plus formatrice pour l'enseignement de l'informatique en CPGE. Cet environnement offre les outils pour une maîtrise complète du système, sans surcouche ni abstraction, ce qui permet aux étudiants de comprendre en profondeur la structure, le fonctionnement et la logique interne d'un système Unix. La gestion du système de fichiers, des utilisateurs, des processus, des périphériques, des droits d'accès ou encore des services en arrière-plan devient visible, manipulable et compréhensible.

C'est aussi une manière de se familiariser avec un écosystème largement répandu dans les domaines de la recherche, de l'ingénierie et de l'administration système.

Ce système respecte la norme POSIX, ce qui garantit une compatibilité avec d'autres systèmes Unix ou de type Unix (comme macOS ou BSD), ce qui en fait un socle pédagogique stable et standardisé. Des distributions comme Ubuntu sont un compromis particulièrement pertinent pour l'enseignement : une interface graphique conviviale, des dépôts de logiciels bien fournis, une large documentation, tout en conservant la puissance et la flexibilité du système Linux sous-jacent. De plus, l'installation d'Ubuntu sur un ordinateur personnel est une étape accessible et bien documentée, ne nécessitant pas de matériel particulier ni de licence payante. Cela permet aux étudiants de travailler dans un environnement homogène, à la maison comme en salle de cours, sans surcoût et sans dépendance à des solutions propriétaires.

Par contraste, une machine virtuelle, bien qu'utilisable dans certains contextes contraints, limite nécessairement l'immersion dans le système. D'abord, les performances sont dégradées : exécuter un système GNU/Linux dans une machine virtuelle implique une surconsommation de ressources (RAM, processeur, disque), ce qui peut conduire à des ralentissements notables, en particulier sur les machines les moins récentes. Ensuite, l'accès au matériel est souvent restreint : la gestion des périphériques réseau, des ports USB ou des fonctionnalités graphiques avancées peut s'avérer partielle, voire impossible à configurer de façon optimale. Enfin, l'environnement virtualisé introduit une couche d'abstraction qui masque de nombreux comportements du système. L'accès aux processus de fond, la configuration réseau réelle ou encore la manipulation directe du système de fichiers sont simplifiés ou artificiellement encapsulés. Cela empêche une appropriation concrète du fonctionnement global d'un système Unix et limite la compréhension fine des interactions entre les différents composants logiciels.

La machine virtuelle reste certes un compromis acceptable dans des environnements où l'on ne peut pas installer Linux directement, par exemple dans le cas d'un matériel imposé, d'un système de parc figé ou d'un besoin de compatibilité temporaire avec certains outils propriétaires. En revanche, elle ne peut en aucun cas remplacer l'expérience complète, directe et authentique qu'offre une installation native. À ce titre, l'installation en dur d'un système GNU/Linux reste la solution la plus cohérente pour former des étudiants à la compréhension du système, à sa maîtrise, et à sa philosophie.

Lorsque des contraintes institutionnelles imposent le maintien d'un système Windows, une solution viable consiste à mettre en place un double démarrage (*dual boot*). Cette configuration permet d'avoir un système GNU/Linux pleinement fonctionnel, sans renoncer à l'accès ponctuel à un

autre environnement. Toutefois, dès que cela est possible, il est préférable d'opter pour une installation Linux exclusive, plus simple à maintenir, plus fluide à utiliser, et surtout plus cohérente pédagogiquement. Elle développe une compréhension authentique de l'informatique, en permettant la manipulation d'un système libre, structuré, documenté, et non un environnement contraint par des choix technologiques extérieurs.

Maîtrise des droits d'accès et apprentissage structuré

Un système GNU/Linux repose sur une gestion rigoureuse des droits d'accès, ce qui en fait un support idéal pour apprendre les principes fondamentaux de la sécurité informatique. L'usage du compte *root* (superutilisateur) permet à l'enseignant de contrôler les installations et les configurations sensibles, tout en laissant aux étudiants la possibilité d'utiliser des commandes, comme `apt-get`, pour demander, ou proposer des installations, dans un cadre maîtrisé. Ce fonctionnement reflète en réalité celui que l'on retrouve dans les environnements professionnels : la séparation des rôles et la traçabilité des interventions.

Richesse des logiciels natifs

L'un des atouts majeurs d'un système d'exploitation libre est la quantité et la qualité des logiciels disponibles directement depuis les dépôts. Sans installation complexe ni conditions de licence restrictives, on peut accéder à une large panoplie d'outils essentiels à l'apprentissage de l'informatique :
– des compilateurs (comme `gcc`, `g++`) ;
– des outils d'automatisation de la compilation (`make`, `cmake`) ;
– des éditeurs puissants (Vim, Emacs, VSCodium) ;
– des interpréteurs (`python`, `bash`, `ocaml`, etc.) ;
– des outils de gestion de versions comme `git`, devenus incontournables dans le développement logiciel moderne ;
– des environnements de développement, des débogueurs, des outils réseau, etc.

Cette richesse est aussi un levier pédagogique : on peut choisir d'introduire progressivement ces outils, les comparer, ou inviter les étudiants à personnaliser leur environnement, ce qu'ils font souvent spontanément, notamment avec des éditeurs comme Emacs, dont la souplesse d'usage et la possibilité de configurer finement les raccourcis, les couleurs, ou les extensions, encouragent une appropriation active de l'environnement de travail.

Git et GitLab : une gestion de versions moderne

Git est un système de gestion de versions décentralisé, conçu en 2005 par Linus Torvalds, le créateur du noyau Linux, pour répondre aux besoins du

développement du noyau. Contrairement à un simple espace de dépôt ou de sauvegarde de fichiers, Git permet de suivre l'évolution d'un projet dans le temps, de revenir à une version antérieure, de comparer des modifications, de travailler à plusieurs en parallèle, et de fusionner des contributions. Ce fonctionnement est devenu la norme dans l'ensemble des projets logiciels modernes, qu'ils soient personnels, collaboratifs, académiques ou industriels.

Pour accompagner l'usage de Git, on peut s'appuyer sur GitLab, une plateforme libre qui permet d'héberger des projets versionnés, de les visualiser, de gérer les droits d'accès, les branches, les demandes de fusion, et même d'automatiser certaines tâches (tests, déploiement, etc.). GitLab peut être installé localement dans un établissement sur un serveur dédié, ou de manière plus souple dans un conteneur *via* Docker, ce qui facilite la maintenance, la portabilité et l'isolation du service. Cette approche par usage de conteneurs permet de disposer d'un environnement entièrement personnalisable, hébergé en interne ou sur un serveur distant, selon les besoins pédagogiques.

Gestion du travail informatique

L'usage de Git n'est pas qu'un outil pratique : il permet une structuration rigoureuse du travail informatique, tant pour les étudiants que pour les enseignantes et les enseignants [1]. Il met en place une logique de versions où chaque modification est enregistrée, contextualisée, datée, et réversible. Les étudiants découvrent ainsi que le développement logiciel ne se résume pas à une série de fichiers finalisés, mais à un processus d'itération, de documentation et de collaboration.

L'apprentissage de Git peut se faire de manière progressive. Dans un premier temps, les étudiants peuvent gérer leurs propres projets en local, en suivant l'évolution de leur travail, en créant des *commits* explicites, en testant des modifications sans crainte de perte. Dans un second temps, ils expérimentent le travail collaboratif, en partageant leurs projets sur GitLab, en gérant des branches, en résolvant des conflits et en découvrant les bénéfices d'un historique partagé.

Organisation claire et gain de temps pour tous

L'utilisation conjointe de Git et GitLab permet de mettre en place une organisation du travail efficace, lisible et reproductible, tant pour les étudiants que pour les enseignantes et les enseignants. Chaque personne dispose d'un dépôt personnel ou collaboratif, accessible depuis n'importe quel poste connecté à Internet. Fini les échanges de fichiers par clé USB ou par e-mail, ou les erreurs de synchronisation : chaque projet est centralisé, versionné et consultable à tout moment. Les pertes de données sont évitées, les erreurs peuvent être retracées, et les différents jalons du projet sont conservés.

Du côté enseignant, GitLab offre de nombreux avantages pédagogiques et logistiques. Il est par exemple très facile de fournir un squelette de projet à l'ensemble des étudiants. Ce dépôt de base peut contenir une arborescence de fichiers, des indications, des exemples ou des contraintes techniques. Chaque étudiant ou binôme peut ensuite bifurquer ce dépôt de référence (*via* une opération de *fork*), créant ainsi son propre espace de travail personnel, tout en conservant un lien clair avec le projet initial. L'enseignante ou l'enseignant peut ainsi structurer le travail en amont, tout en conservant une vision globale de l'avancement et des choix techniques de chacun.

De plus, pour les rendus, il suffit de cloner les dépôts étudiants pour accéder à leur dernière version à une date donnée, sans avoir à gérer des fichiers transmis hors délai ou dans des formats inattendus. L'environnement de développement est reproductible, ce qui facilite l'évaluation, la relecture, la comparaison, ou même la démonstration d'un projet en classe. GitLab permet également de suivre l'activité individuelle de chaque étudiant dans un projet collaboratif (nombre de commits, type de modifications, etc.), offrant ainsi un support objectif d'évaluation continue.

Enfin, l'usage de Git incite naturellement à privilégier des fichiers texte simples : code source, fichiers de configuration, documentation, README, etc. Cette approche encourage la clarté, la sobriété, la compatibilité entre systèmes et prépare aux bonnes pratiques du développement logiciel. Le format Markdown, très utilisé pour la documentation dans les dépôts, s'intègre parfaitement dans cette logique et sera développé dans la section suivante.

Démarche pédagogique active et professionnelle

Git habitue les étudiants à une méthodologie professionnelle de développement : gestion du temps, clarté des modifications, documentation intégrée, et respect d'un environnement partagé. Il prépare aux exigences des projets en école d'ingénieurs, en recherche ou en entreprise, tout en posant les bases d'un travail reproductible, traçable et rigoureux.

Du point de vue pédagogique, l'introduction de Git dès les premiers mois de la formation en MP2I et MPI permet d'instaurer une culture du code claire et structurée, dans laquelle chaque étudiant développe des compétences transversales : lecture de l'historique, relecture de code, structuration des *commits*, interactions dans un espace partagé.

HedgeDoc : un environnement de prise de notes collaboratif et structurant

HedgeDoc est un éditeur de texte collaboratif en ligne, conçu pour permettre à plusieurs utilisateurs d'écrire simultanément dans un document au format

Markdown. Il permet une visualisation instantanée du rendu final en parallèle du code source. Cette interaction directe entre le contenu brut (Markdown) et sa mise en forme constitue un atout pédagogique fort : elle rend les mécanismes de structuration du texte immédiatement visibles, tout en permettant de rester concentré sur l'essentiel, le contenu, et non l'interface.

Dans le cadre de l'enseignement en MP2I et MPI, HedgeDoc peut être installé sur un serveur accessible à l'ensemble des étudiants, selon un principe similaire à GitLab. Une installation par conteneur Docker permet de déployer facilement le service sur un serveur, tout en assurant une accessibilité continue depuis la salle de classe comme depuis un poste personnel. Cette installation autonome permet de se détacher de solutions propriétaires externes et d'offrir un environnement maîtrisé, sécurisé, et conforme aux principes du libre.

Prise des notes autrement

L'un des usages les plus pertinents de HedgeDoc est la prise de notes collaborative [2], notamment en cours, en travaux dirigés ou en travaux pratiques. Le programme d'informatique en MP2I s'y prête particulièrement bien, même si cela nécessite un travail préalable d'organisation et de préparation de la part des enseignantes et des enseignants. Lorsqu'elle est possible structurellement, cette pratique apporte de nombreux bénéfices pédagogiques. Les étudiants peuvent travailler en binôme ou en petit groupe, synchroniser leurs apports, se corriger mutuellement, construire une synthèse commune. Cette dynamique de collaboration active renforce l'engagement en classe et développe des compétences précieuses : écoute, clarté, esprit de synthèse, précision de l'écriture.

D'un point de vue plus technique et pragmatique, l'utilisation régulière de HedgeDoc permet aux étudiants de développer leur aisance au clavier, une compétence encore inégalement acquise à l'entrée en CPGE. Écrire sans chercher les touches, structurer un texte à la volée, insérer du code ou des formules, reformuler un contenu entendu ou compris : autant de gestes qui s'affinent avec la pratique, et que l'outil favorise.

Markdown et Latex : pour une écriture simple, sobre et efficace

L'un des principes fondateurs de HedgeDoc est son utilisation exclusive du format Markdown, un langage de balisage léger conçu pour structurer un texte sans interface graphique lourde. Au lieu de chercher des options dans une barre d'outils, les étudiants apprennent à utiliser des symboles simples ("#" pour les titres, "∗" pour les listes, etc.), ce qui renforce la maîtrise du

contenu et de sa structure. Cette pratique sensibilise également à l'intérêt des fichiers de texte simples et portables, au détriment des formats binaires fermés.

Markdown peut être enrichi de manière fluide par des éléments Latex, permettant d'intégrer des formules mathématiques directement dans le document. Cela ouvre la voie à une écriture scientifique lisible, propre et exportable, sans avoir recours à des traitements de texte complexes.

Outil à usage pédagogique personnel

Les étudiants prennent généralement en main HedgeDoc très rapidement, grâce à son interface claire et à l'édition en temps réel. Ils l'adoptent aussi bien comme outil de travail collaboratif que comme support personnel pour organiser leur apprentissage. Au fil des séances, il devient un véritable espace de production et de structuration des connaissances.

Ils y construisent leur propre documentation : synthèses de cours, réponses à des exercices, rappels de syntaxe, résumés de méthodes, bibliothèques de commandes utiles, etc. Le caractère textuel et brut de Markdown encourage une écriture sobre, lisible, et facilement versionnable (notamment en lien avec Git). Ils apprennent à organiser leurs notes de manière hiérarchisée, à distinguer l'essentiel de l'accessoire, et à enrichir progressivement un document qu'ils peuvent réutiliser ou adapter dans d'autres contextes.

Par ailleurs, HedgeDoc peut être utilisé pour préparer des exposés, servir de journal de bord de projet, ou encore pour les TIPEs. Les étudiants peuvent y inclure des explications, des fragments de code, des erreurs rencontrées, ou des pistes de réflexion ou construire des fiches récapitulatives. Ces pratiques favorisent non seulement l'autonomie, mais aussi le développement de compétences rédactionnelles et de métacognition.

Conclusion

L'utilisation conjointe de GNU/Linux, GitLab et HedgeDoc dans le cadre de l'enseignement de l'informatique en MP2I et MPI constitue une démarche cohérente, à la fois sur le plan pédagogique, technique et éthique. Elle transforme concrètement les pratiques pédagogiques en structurant des séquences d'apprentissage autour de projets techniques authentiques : versionner un code source avec Git, rédiger une documentation collaborative en Markdown, automatiser des tests, ou encore configurer un environnement Linux complet. Ces activités, loin d'être accessoires, deviennent le support même des apprentissages. Les étudiants développent ainsi des compétences transversales : organisation de projet, compréhension de l'historique des erreurs, expérimentation autonome, appropriation de la documentation technique. Cette approche rend visibles les étapes du raisonnement, invite à documenter ses

choix, à structurer ses méthodes de travail, et à entrer dans une culture informatique ancrée dans la pratique réelle. Elle fait émerger une posture active et critique, qui ne sépare pas l'outil de la pensée qu'il permet. Ces outils libres proposent ainsi un environnement d'apprentissage ouvert, structuré et rigoureux, qui permet aux étudiants de se former en profondeur aux pratiques réelles de l'informatique.

Au-delà des avantages fonctionnels, cette infrastructure logicielle présente également un intérêt majeur en termes de protection des données personnelles. Contrairement à de nombreuses solutions propriétaires, GitLab et HedgeDoc, lorsqu'ils sont installés sur un serveur local ou administré par l'établissement, stockent l'intégralité des données produites sur ledit serveur, ces dernières ne quittent pas le cadre institutionnel. Du côté du système d'exploitation, GNU/Linux présente également des garanties importantes. Les comptes utilisateurs sont entièrement locaux, sans identification centralisée par des services extérieurs. Ces outils libres ne sont ni soumis à des politiques d'exploitation commerciale, ni exposés aux réglementations extraterritoriales comme le *Cloud Act* américain, qui peut contraindre certaines entreprises à fournir des données à des autorités étrangères. Par exemple, un compte Microsoft lié à des services en ligne, soulève des questions de confidentialité dans un cadre éducatif. L'usage d'outils libres et auto-hébergés assure ainsi une conformité au RGPD et une maîtrise complète de l'environnement numérique, en accord avec les principes de souveraineté numérique et de protection des étudiants.

Cette démarche ne peut toutefois reposer uniquement sur l'enthousiasme individuel des enseignants. Elle suppose un engagement institutionnel fort. Il est donc essentiel que les besoins spécifiques de l'enseignement de l'informatique en CPGE soient pleinement intégrés aux politiques éducatives, tant au niveau académique que ministériel. Cela implique de garantir des moyens techniques adaptés (serveurs, salles équipées, connexions stables), de reconnaître officiellement la nécessité de disposer d'installations Linux en natif dans les salles de travaux pratiques, et de soutenir les équipes pédagogiques par l'affectation de personnels qualifiés en administration système et réseau. Plus largement, une politique publique ambitieuse en faveur du logiciel libre dans l'éducation, accompagnée de dispositifs de formation continue pour les enseignants et les personnels techniques, permettrait d'assurer une diffusion équitable et durable de ces pratiques. En intégrant ces outils dans les pratiques pédagogiques, on ne se contente donc pas d'enseigner l'informatique : on enseigne aussi une manière éthique, responsable et lucide de la pratiquer.

Références

[1] Julio César Cortés Ríos, Kamilla Kopec-Harding, Sukru Eraslan, Christopher Page, Robert Haines, Caroline Jay, et Suzanne Embury. 2019. A Methodology for Using GitLab for Software Engineering Learning Analytics. 3-6. https://doi.org/10.1109/CHASE.2019.00009.

[2] Axel Dürkop. 2022. *Collaborative content creation with HedgeDoc*. (Consulté le 17 juillet 2025). Technical University of Hamburg (TUHH). Consulté à l'adresse http://hdl.handle.net/11420/14416.

[3] IGÉSR. 2024. La préparation aux formations et aux métiers du numérique et de l'informatique : parcours, programmes, pédagogie, mixité des cursus dans les lycées généraux et technologiques et dans les lycées professionnels. (Rapport nº 22-23-006A).

[4] Ministère de l'Éducation nationale de la Jeunesse et des Sports. Programme d'informatique de MP2I et MPI. (Consulté le 17 juillet 2025). Consulté à l'adresse https://www.education.gouv.fr/bo/21/Special1/ESRS2035777A.htm.

[5] Richard M. Stallman. 2002. *Free Software, Free Society: Selected Essays of Richard M. Stallman*. (Consulté le 17 juillet 2025). GNU Press, Boston, MA. Consulté à l'adresse https://www.gnu.org/philosophy/fsfs/rms-essays.pdf.

[6] Linus Torvalds et David Diamond. 2001. *Just for Fun: The Story of an Accidental Revolutionary*. HarperBusiness.

La grande muraille d'Égypte

Une histoire de maçons et de parallélisme…

Emmanuelle Saillard
Inria

L'histoire de la grande muraille d'Égypte commence en l'an −51 lorsque la reine Cléopâtre règne sur l'Égypte. Afin de montrer la grandeur de l'Égypte face à l'empire romain, Cléopâtre ordonne la construction d'une muraille autour de son palais. Comme elle habite en Égypte, elle aimerait sa muraille en forme de triangles les uns à côté des autres pour rappeler les pyramides. Pour réaliser cette tâche, elle fait appel à Numérocis, son meilleur maçon. Elle lui demande d'utiliser des briques colorées qui ont la particularité d'être jolies mais très lourdes. N'ayant pas de couleur préférée, elle laisse Numérocis libre sur le choix des couleurs. La seule contrainte imposée par Cléopâtre est de réaliser la muraille le plus rapidement possible et avec deux couleurs. Devant l'ampleur de la tâche, Numérocis demande à son ami Numérodis de l'aider. Afin de vérifier l'évolution de la construction, Cléopâtre charge Pénaltis, son maître des pénalités, de distribuer des pénalités si un maçon ne travaille pas suffisamment.

Le but de cette activité est d'introduire un domaine de l'informatique qui s'appelle le calcul haute performance, à travers la construction d'un morceau de la muraille triangulaire. À la fin de l'activité, les joueurs auront un aperçu de certaines difficultés liées à l'écriture de programmes parallèles.

Le public

Cette activité est recommandée pour des élèves de collège et lycée mais peut être adaptée pour des élèves en école primaire.

DOI:10.48556/SIF.1024.26.35

Le matériel

Pour réaliser l'activité, il est nécessaire de disposer de cubes de deux couleurs différentes qui formeront deux *tas de briques* (petits cubes en bois, legos, carrés de sucre…). Il faut également imprimer les fiches personnages, les jetons pénalités et les cartes, donnés plus loin dans cet article. Il est recommandé de plastifier les cartes et les jetons pénalités pour faciliter leur manipulation. Dans le reste de l'article, nous supposerons avoir des briques jaunes et bleues.

Le contexte

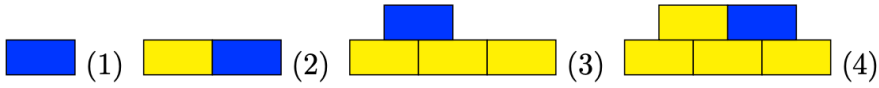
L'activité se joue de préférence par groupe de trois car il y a trois personnages : deux maçons (Numérocis et Numérodis) et un maître des pénalités (Pénaltis). Chaque groupe de participants a des cartes à sa disposition qui sont de trois types : carte « réflexion », carte « construction » et carte « savez-vous ? » :

– les cartes « réflexion » cachent des mathématiques ! L'idée est de retrouver des formules mathématiques ;
– les cartes « construction » proposent de construire un motif qui est donné, ou bien à créer ;
– les cartes « savez-vous ? » ont pour but d'informer sur différents sujets liés au calcul haute performance.

Les maçons doivent reproduire à l'identique le motif qui se trouve sur la carte piochée ou en créer un. Pour chaque construction, les maçons doivent respecter les deux règles d'or suivantes :

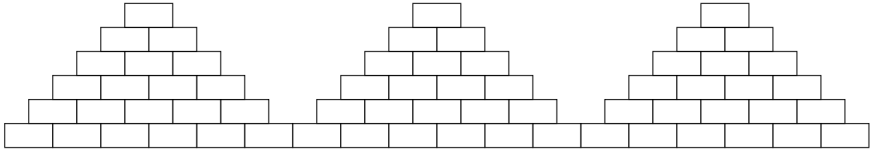
– les maçons choisissent chacun une couleur de brique différente avant de commencer la construction et ils ne poseront chacun que des briques de cette couleur ;
– un maçon peut poser une brique si les conditions suivantes sont réunies (dans les schémas ci-dessous, la brique qui va être posée est la brique bleue, et tous les cas de figure sont représentés sur ce schéma) :

  → la brique sera posée sur le sol, ou bien à cheval sur deux briques déjà posées au niveau en dessous (comme le montrent tous les schémas ci-dessous),
  → la brique sera posée à la position la plus à gauche du niveau considéré (schémas 1 et 3) ou bien immédiatement à droite d'une brique déjà posée (schémas 2 et 4).



Pénaltis est responsable de l'évaluation de la construction. Il vérifie que la construction se fait dans les temps. La construction prend du retard quand un même maçon pose deux briques de suite (et l'autre doit attendre). Pénaltis posera un jeton *pénalité* dans ce cas (on a deux briques bleues de suite, ou deux briques jaunes de suite). Selon le motif choisi pour la construction, un maçon pourra être amené à attendre pour poser sa brique.

Comme on pourra le voir sur les premières cartes « construction », les maçons ont comme mission de construire le début de la muraille représentant 3 triangles en tout, comme sur la figure suivante :



Le but est de construire chaque triangle de la muraille le plus efficacement possible. Chaque triangle a 6 briques pour la base et se construira en utilisant les cartes « construction ». Les constructions se feront avec deux couleurs qui montrent la répartition du travail entre Numérocis et Numérodis, les deux maçons responsables de la construction de la muraille. Pour plus d'originalité, les maçons construiront différents motifs sur chaque triangle. Un motif est défini par la coloration des briques.

Le déroulé

Comptez environ 50 minutes pour faire l'activité. Pour une classe, faire des groupes de 3 élèves si possible (un groupe de deux est possible, un des maçons jouera alors le rôle du maître des pénalités). Chaque groupe aura un tas de cartes, les fiches personnages et des briques.

L'activité se joue en piochant les cartes les unes après les autres. Faites un tas avec les cartes, faces visibles ou cachées, en les mettant dans l'ordre.

Jouez en piochant les cartes une à une, jusqu'à ce que le tas soit vide. Une partie simple comprend 7 cartes. Selon le temps et la motivation des joueurs, des cartes bonus sont en option sur ce site[1].

Chaque carte « construction » construit un des triangles de la muraille triangulaire. La première carte est une carte « construction » qui a pour but de se familiariser avec les règles des personnages. Elle propose un motif à construire pour le premier triangle de la muraille. Pour ce motif, Pénaltis posera forcément des pénalités. Les cartes 3 et 6 demandent respectivement de construire deux nouveaux motifs : un avec le plus de pénalités possibles et un avec le moins de pénalités possibles. Plusieurs motifs existent dans les deux cas.

La deuxième carte est une carte « réflexion »qui propose de retrouver la formule donnant la somme des x premiers entiers. L'idée est de mettre les briques sous forme d'un demi rectangle et de voir qu'un autre demi rectangle peut s'emboiter à côté pour former un rectangle entier. La taille du rectangle et une réflexion autour de la quantité de briques aide à retrouver la formule. La carte 4 rappelle des notions de probabilités. Pour trouver combien de motifs existent avec N briques, il faut trouver combien de possibilités on a pour poser la première brique, puis la deuxième, puis la troisième, etc…

La carte « savez-vous ? » numéro 5 fait le lien entre l'activité et plus précisément une construction de muraille et le calcul haute performance. La dernière carte (numéro 7) interroge sur une expression largement utilisée de nos jours. Un indice : ce n'est pas Mr Bug !

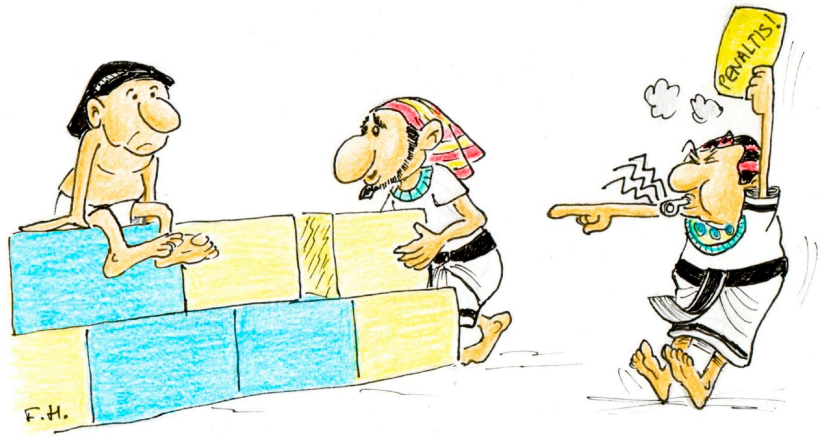
Retour à l'informatique

La construction de la muraille est comme un programme à exécuter. Chaque bout de mur triangulaire représente un calcul et le motif sur ce bout de mur met en avant la répartition du calcul entre les deux maçons. Si Numérocis construit tout seul la muraille à motifs triangulaires, la construction prendra du temps. C'est ce qu'il se passe dans la carte 3. Faire appel à Numérodis va nous permettre de construire la muraille en moins de temps. En effet, tant qu'on n'a pas de pénalité, le fait d'avoir deux maçons permet à Numérodis de poser une brique pendant que Numérocis fait son aller-retour pour en poser une autre, sans le ralentir. Cette construction en parallèle donne un des motifs possibles demandé sur la carte 6 et divise le temps de construction par 2.

Pour retranscrire le gain de temps pour une construction, Pénaltis évalue celle-ci en posant des jetons *pénalité*. Certains motifs entrainent forcément des pénalités, tout comme les calculs dans un programme ont souvent des dépendances (une étape d'un calcul doit être faite avant une autre étape). La

1. https://emmanuellesaillard.fr/mediation/.

première construction demandée sur la carte 1 en est l'exemple. Le nombre de pénalités dépend de la façon dont les maçons se sont organisés (Numérocis peut faire en sorte de poser le plus de briques possibles pour que Numérodis puisse poser sa première brique le plus vite possible) et du motif à réaliser. Pénaltis estime qu'une construction est optimale si les maçons posent une brique à tour de rôle (carte 6).



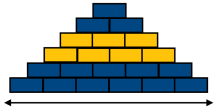
Le calcul haute performance permet d'effectuer rapidement des calculs complexes et des traitements de données massives. La construction de la muraille met en avant l'importance de bien réfléchir au découpage d'un programme parallèle pour qu'il s'exécute le plus efficacement possible (c'est-à-dire rapidement). Les constructions se font avec deux couleurs pour distribuer la charge de travail entre Numérocis et Numérodis. C'est la même chose lorsqu'on veut répartir du temps de calcul entre deux processeurs. Le découpage du calcul est appelé un motif dans l'activité. Un motif retranscrit la difficulté du découpage d'un calcul dans certains cas. Après l'exécution d'un programme, le développeur ou la développeuse analyse généralement les résultats obtenus pour s'assurer que le programme s'exécute correctement et cherche à améliorer les performances de celui-ci (c'est-à-dire réduire le temps d'exécution). Pour cela, on doit identifier les parties du programme qui prennent beaucoup de temps. Ce sont les briques où l'on a eu des pénalités.

Les femmes aussi font de l'informatique ! Peu de gens le savent mais les femmes sont à l'origine de plusieurs concepts informatique. C'est le cas de Grace Hooper qui est réputée avoir utilisé le mot « *bug* » pour la première fois. Ce mot est encore utilisé aujourd'hui pour parler d'une erreur dans un programme.

Les fiches personnages et cartes à imprimer[2]

1 CARTE CONSTRUCTION

Construire le motif ci-dessous pour le 1er triangle. Maçons, attention à bien respecter les règles d'or !



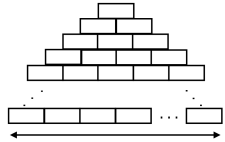
Base = 6 briques

Pénaltis annoncera le nombre de pénalités à la fin de la construction.

2 CARTE RÉFLEXION

Pouvez-vous retrouver la formule mathématique qui donne le nombre de briques sur la façade à construire en fonction du nombre de briques à la base ?

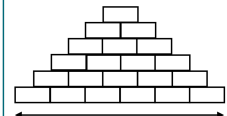
Aide : Commencez avec une base 4 (x=4) et placez toutes les briques plus à gauche sur chaque étage



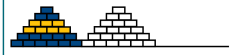
Base = x briques

3 CARTE CONSTRUCTION

Pour le 2e triangle, dessiner un motif qui d'après vous a une pénalité **la plus grande possible** et construisez-le. Pénaltis annoncera le nombre de pénalités à la fin de la construction.



Base = 6 briques

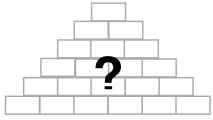


PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ

2 CARTE RÉFLEXION

Combien y a-t-il de motifs possibles avec 2 couleurs et N briques au total ?

Aide : Faites tous les motifs possibles avec une base 2 et comptez combien vous en avez !



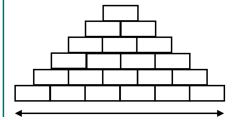
N = nombre total de briques

5 SAVEZ-VOUS … ?

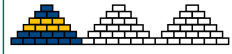
D'après-vous qu'est-ce que le **parallélisme en informatique** ?

6 CARTE CONSTRUCTION

Pour le 3e triangle, dessiner un motif qui d'après vous a une pénalité **la plus petite possible** et construisez-le. Pénaltis annoncera le nombre de pénalités à la fin de la construction.

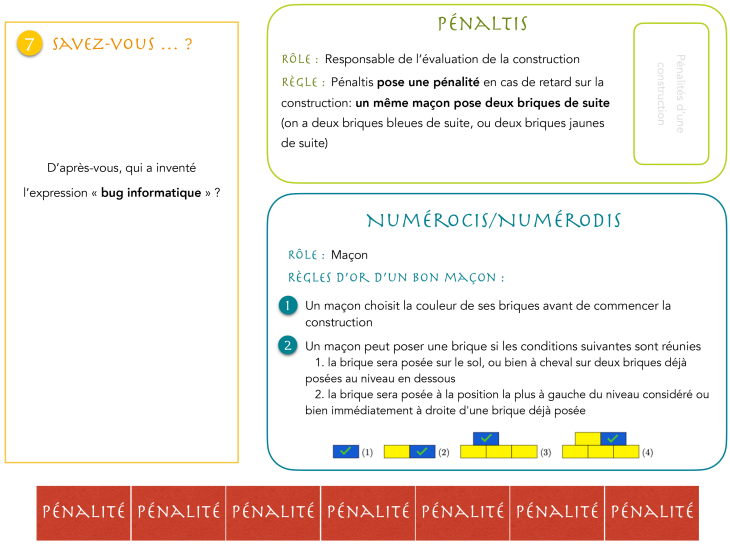


Base = 6 briques



PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ PÉNALITÉ

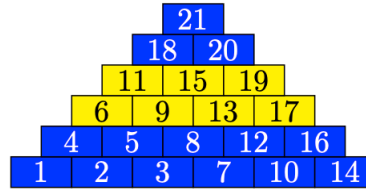
2. Fiches à télécharger ici : `https://1024.socinfo.fr/2025/12/cartes.pdf`.



Réponses aux questions des cartes

1. Carte « construction »

Le motif du premier triangle entraîne forcément un nombre de pénalités non nul. Le nombre total de pénalités dépend de la façon dont les maçons s'organisent (Numérocis peut faire en sorte de poser le plus possible de briques pour que Numérodis puisse poser sa première brique le plus vite possible). Pour obtenir le nombre minimum de pénalités (6), les briques doivent être posées dans cet ordre :

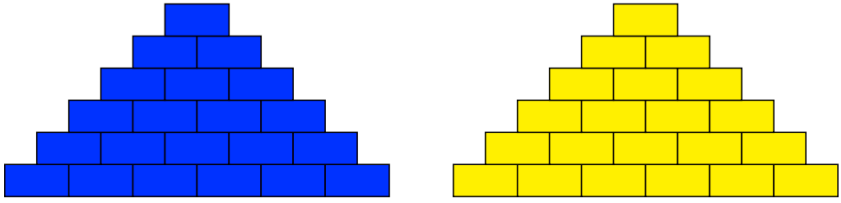


2. Carte « réflexion »

Le nombre de briques pour une base de x briques est $\sum_{i=1}^{x} i = \frac{x \times (x+1)}{2}$. La démonstration peut se faire par récurrence. Pour retrouver la formule, on peut mettre les briques sous forme d'un demi rectangle et remarquer qu'un autre triangle peut s'emboiter pour former un rectangle de taille $x \times (x+1)$. Le rectangle contient deux fois plus de briques que le triangle, il faut donc diviser la taille du rectangle par 2.

3. Carte « construction »

Un motif avec une pénalité la plus grande possible est l'exemple d'un travail très mal réparti entre les maçons. Un seul maçon travaille, la construction est séquentielle. Les deux motifs possibles sont les suivants (santionnés par 20 pénalités) :



4. Carte « réflexion »

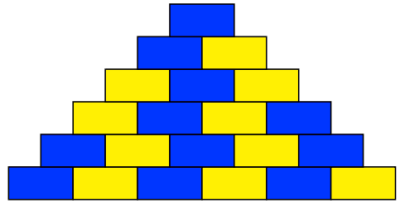
Il y a 2^N motifs possibles avec deux couleurs. Pour chaque brique, on a deux choix de couleur possibles.

5. Carte « savez-vous ? »

Le parallélisme désigne le fait, pour un système, d'être capable d'effectuer plusieurs unités de calcul simultanément pour accélérer l'exécution d'une application.

6. Carte « construction »

Un motif avec une pénalité la plus petite possible est l'exemple d'un travail parfaitement réparti entre les maçons. Un exemple de motif est (sans aucune pénalité) :



7. Carte « savez-vous ? »

Une légende raconte que l'expression « bug informatique » a été utilisée pour la première fois en 1947 par Grace Hopper, informaticienne, mathématicienne et officier supérieur de la marine américaine. Après avoir découvert un papillon de nuit dans la machine Mark II, l'équipe de Grace Hopper a décidé de scotcher l'insecte dans le journal de bord de l'ordinateur avec la mention manuscrite « *first actual case of bug being found* » (que l'on pourrait traduire par « première découverte d'un véritable *bug* »).

Réseaux de Petri dans le premier degré

Alain Busser, professeur de NSI
Lycée Roland-Garros du Tampon, IREMI de La Réunion

Dans cet article, on relate une expérience de médiation menée dans des écoles et collèges (essentiellement en cycle 3) lors des fêtes de la science 2019 et 2020, autour d'un artefact de calcul peu connu : les réseaux de Petri.

Définitions

Réseau de Petri — Places et transitions

Un *réseau de Petri* est un graphe orienté biparti, c'est-à-dire un dessin comme celui de droite, où des *sommets* (ronds ou carrés) sont reliés par des *arcs* (des flèches).

Ci-contre le graphe est parfois qualifié de « multigraphe » parce qu'il peut y avoir plusieurs arcs reliant deux sommets donnés.

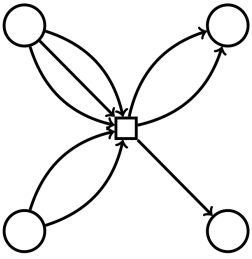


Fig. 1. Un réseau de Petri.

Les sommets sont de deux sortes :

- ◯ les sommets ronds s'appellent des *places* ;
- □ les sommets rectangulaires (ici, carrés) s'appellent des *transitions*.

Graphe biparti

Que le graphe soit biparti se traduit ici par le fait qu'aucun arc ne va d'une place à une place ni d'une transition à une transition. Tout arc va :

– ou bien d'une place vers une transition ;
– ou bien d'une transition vers une place.

DOI:10.48556/SIF.1024.26.45

Un premier jeu abordable dès le cycle 3, consiste simplement à dessiner un réseau de Petri, c'est-à-dire un graphe vérifiant les propriétés ci-dessus. Mais un réseau de Petri est un objet statique, et lorsque Carl Adam Petri a créé cette notion en 1962 [4], ce n'était pas seulement pour les dessiner, mais pour jouer avec [2]. Pour cela il faut des *jetons*.

Réseau de Petri marqué

Un *marquage* d'un réseau de Petri, consiste à placer dans certaines places, des jetons. Par exemple, on voit sur la figure 2 un marquage du réseau de Petri précédent, par 4 jetons.

Les jetons sont dessinés l'un à côté de l'autre pour mieux les dénombrer, mais avec les élèves, on a utilisé des jetons plats et empilés. Reste à voir comment les manipuler.

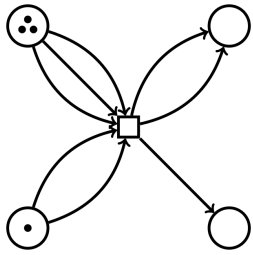


Fig. 2. Un réseau de Petri marqué.

Manipulation des jetons

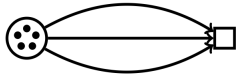
Le mot *jeton* suggère une activité de médiation basée sur les réseaux de Petri ; elle nécessite le matériel suivant :

- une feuille format A4 par réseau de Petri (le réseau pouvant être imprimé avant l'activité, ou dessiné par les élèves eux-mêmes, ils adorent ça) ;
- des jetons (graines ou jetons plats style poker) en quantité suffisante pour occuper les places au cours de l'activité ;
- un espace (par exemple une boîte) pour stocker les jetons qui ne sont pas sur le réseau de Petri ;
- (en cycle 4) un tableau pour noter les marquages initiaux et finals d'un réseau de Petri (afin d'émettre des conjectures).

Le projet initial était de tester l'apprentissage de problèmes du champ additif par les réseaux de Petri. Par exemple le problème suivant :

> *Lucie a perdu 3 billes à la récréation. Il lui reste 5 billes. Combien de billes avait-elle avant la récréation ?*

se modélise par le réseau de Petri marqué ci-contre (voir plus bas pour le pourquoi du comment : le problème est évidemment de reconstituer l'état initial du réseau, c'est-à-dire le marquage qui précédait celui que l'on voit ci-contre).



Les calculs que l'on fera plus bas avec des réseaux de Petri utilisent la numération unaire (l'entier n est codé par n jetons) ce qui permet à des élèves ne connaissant pas la numération, de mener ces calculs quand même.

Le premier public visé était composé d'élèves de cycle 2 ou de classes ULIS. Un concours de circonstances (parmi lesquelles un certain coronavirus) a empêché de toucher ce public. Ce sont donc des élèves des cycles 3 et 4 qui ont bénéficié de l'activité. Cela a permis de les familiariser avec les graphes, l'arithmétique et les systèmes d'équations linéaires, ainsi que les notions d'algorithmes, terminaison des algorithmes et invariants.

Places et transitions

Les places du réseau de Petri s'appellent ainsi, parce que c'est là que l'on place des jetons. Pour le marquage initial, mais aussi lors des différentes étapes du calcul.

Les transitions s'appellent ainsi parce que les jetons ne font qu'y transiter. Dans l'exemple de la figure 2 ci-dessus, on constate que la transition a un degré entrant égal à 5 (il y a 5 flèches qui arrivent vers cette transition) mais un degré sortant égal à 3 seulement (3 flèches sortent cette transition). Comme on le verra un peu plus loin, il y aura une perte nette de $5 - 3 = 2$ jetons lorsque l'on déclenchera la transition. Pour gérer ce phénomène (possibilité que des jetons soient perdus ou ajoutés lors de la transition), on dispose une boîte contenant beaucoup de jetons entre les joueurs. Par exemple sur une table carrée, avec 4 joueurs disposés aux points cardinaux, on dispose cette boîte au centre de la table. On appelle *banque de jetons* cette boîte. Alors, si le déclenchement d'une transition requiert plus de jetons qu'il y en a, on puise les jetons supplémentaires dans la banque, et s'il y a des jetons à retirer du réseau de Petri, on les place dans la banque.

Le réseau de Petri de la figure 2 est bloqué (on dit qu'il est *mort*) parce que son unique transition ne peut être déclenchée. Pour que la transition puisse être déclenchée (on dit alors qu'elle est *vivante*), il faut qu'il y ait au moins 3 jetons dans la place en haut à gauche (autant que de flèches partant de la place vers la transition, c'est le cas ici), mais aussi au moins 2 jetons dans la place en bas à gauche. Or ici il n'y a qu'un jeton, ce qui suffit à bloquer la transition.

Déclenchement d'une transition

Le marquage de la figure 4, partie de gauche, en revanche, permet à la transition de se déclencher.

En effet pour que la transition puisse se déclencher, il faut au moins 3 jetons dans la place en haut à gauche (or $5 \geqslant 3$) et 2 jetons dans la place en bas à gauche (or $6 \geqslant 2$). Dans la suite de cet article, on convient de colorier

les transitions actives. S'il y a au moins une transition qui peut être déclenchée, le réseau de Petri est dit *vivant*.

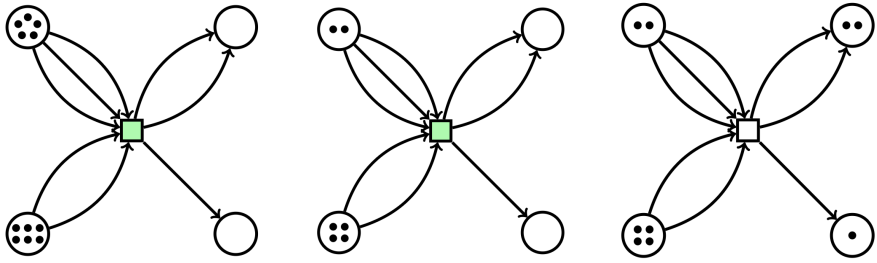


Fig. 4. De gauche à droite : marquage initial, transition à moitié tirée, transition tirée.

Le déclenchement d'une transition se fait en deux temps :
– tout d'abord, un jeton glisse le long de chacun des arcs entrants pour aller dans la banque (résultat sur le réseau au milieu de la figure 4) ;
– ensuite, des jetons sont prélevés de la banque pour glisser le long des arcs sortant de la transition, pour se placer dans les places en aval (résultat sur le réseau à droite de la figure 4).

On constate qu'il y a moins de 3 jetons dans la place en haut à gauche et la transition n'est donc plus active : le réseau de Petri est mort, et le calcul s'est achevé. Le résultat du calcul est, ici, égal à $(2, 1)$, que l'on lit sur les deux places à droite sur le réseau.

Initiation à l'algèbre linéaire

En cycle 4, on peut faire jouer plusieurs élèves sur le même réseau de Petri (cela a été fait lors de la semaine des maths, en séances d'une demi-heure ou d'une heure selon le collège) et faire remplir un tableau, où, en appelant a et b les nombres de jetons dans les places de gauche, c et d celles de droite, on sait que $a \geqslant 3$ et $b \geqslant 2$ (sinon on a automatiquement $c = d = 0$) et l'exemple ci-dessus amène à cette ligne du tableau :

a	b	c	d
5	6	2	1
…	…	…	…
…	…	…	…

La question est alors de trouver deux fonctions f et g de deux variables (deux expressions en a et b) telles que

$$c = f(a,b) \qquad \text{et} \qquad d = g(a,b) \tag{1}$$

Avec la moitié d'une classe de 3^e, en répartissant bien les marquages initiaux entre élèves, on a en quelques minutes une quinzaine de lignes dans le tableau précédent, ce qui amène alors à l'émission de conjectures sur l'expression algébrique de f et g.

Calcul par réseaux de Petri

Un réseau de Petri peut très bien ne jamais mourir, comme par exemple celui-ci :

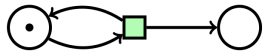


Fig. 5. Réseau de Petri immortel.

En effet, comme la transition réinjecte un jeton dans la place de gauche, elle est éternellement vivante et elle va éternellement injecter un jeton dans la place de droite.

Mais lorsqu'un réseau de Petri s'arrête au bout d'un temps fini, certaines places (en général initialement vides) contiennent un nombre de jetons qui dépend du nombre de jetons dans les places du départ, et le réseau de Petri a calculé les nombres d'arrivée, qui sont fonction des nombres de départ. On peut alors (se) poser deux questions :
– étant donné un réseau de Petri, que calcule-t-il ?
– étant donné une fonction, quel réseau de Petri la calcule ?
Ci-après on va voir comment effectuer les 4 opérations avec des réseaux de Petri, mais pour la multiplication et la division on aura besoin d'arcs inhibiteurs.

Addition

Un réseau de Petri effectuant une addition doit avoir au moins trois places :
– une pour le premier terme de la somme ;
– une pour le second terme ;
– et une pour la somme ;
et au moins deux transitions (une par terme). Le réseau de Petri de la figure 6 est donc minimal.

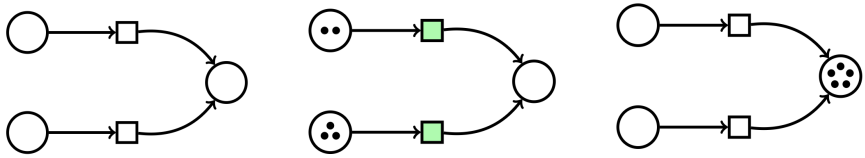


Fig. 6. Un réseau de Petri effectuant une addition.

Pour calculer $2 + 3$, on place respectivement 2 jetons et 3 jetons dans les places de gauche (voir le réseau au centre de la figure 6). Les deux transitions sont vivantes, on a donc le choix entre déclencher celle du haut ou déclencher celle du bas. Chaque fois qu'on déclenche une transition, cela a pour effet de transférer un jeton depuis sa place d'origine, vers la place d'arrivée. Ceci peut être fait dans l'ordre qu'on veut[1], et lorsque plus aucune des transitions ne peut être déclenchée, on arrive à la fin de l'addition (voir le réseau à droite sur la figure 6).

Soustraction

Le réseau de Petri ci-dessous (figure 7) à gauche, permet d'effectuer une soustraction.

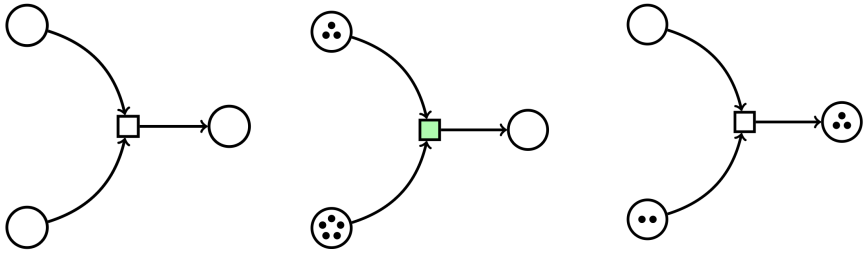


Fig. 7. Un réseau de Petri effectuant une soustraction.

En effet si on le marque avec respectivement 3 et 5 jetons (image centrale de la figure 7), alors il finit son calcul avec 2 jetons dans la place du bas (image de droite de cette même figure). Cette disposition permet de savoir :
– que $|5 - 3| = 2$ (le nombre de jetons dans la place de gauche) ;

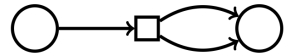
1. On dit qu'un tel réseau de Petri est *asynchrone*.

– que 5 était le plus grand des deux nombres (la place non vide est celle du bas, qui contenait initialement 5 jetons) ;
– et que le plus petit des deux nombres est 3 (nombre de jetons dans la place de droite).

Ce réseau est recyclé dans la partie suivante, la soustraction intervenant dans la division euclidienne ainsi que dans l'algorithme d'Euclide.

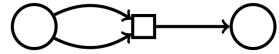
Vers le binaire

Voici un réseau de Petri doubleur :

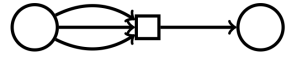


Si on l'initialise avec n jetons dans la place de gauche, il finira avec $2 \times n$ jetons dans la place de droite. Il a été construit à partir de l'additionneur en fusionnant les deux termes.

On peut inverser les rôles du nombre et son double :



ce qui donne un réseau de Petri diviseur par 2. La division en question est euclidienne (il y a un reste dans la place de gauche à la fin du calcul, et le quotient est dans la place de droite). On peut le généraliser à un réseau de Petri diviseur par 3 :



En multipliant des étages identiques, on aboutit à un convertisseur unaire→ binaire :

c'est-à-dire que si on place n (par exemple 5) jetons dans la place de droite :

le calcul se termine avec la représentation binaire de n (ici 101) (qui se lit de gauche à droite) :

Ce réseau de Petri (en fait la version à 4 places qui convertit en binaire les nombres allant de 0 à 15) a eu beaucoup de succès en cours moyen, probablement parce qu'il faut un temps raisonnable pour effectuer les 16 conversions. On peut l'étendre de manière à modéliser l'abaque binaire de Neper comme un réseau de Petri [3].

Réseaux de Petri à arcs inhibiteurs

Un arc (ou flèche) allant d'une place à une transition a pour effet de permettre la transition s'il y a au moins un jeton en amont de l'arc. Un *arc inhibiteur* (représenté par une pointe de flèche circulaire) a l'effet contraire : la transition ne peut se déclencher que s'il n'y a *pas* de jeton en amont. Pour qu'une transition soit vivante il faut alors :

– qu'en amont de chaque arc venant vers la transition, il y ait au moins un jeton ;
– qu'en amont de tous les arcs inhibiteurs, les places soient vides.

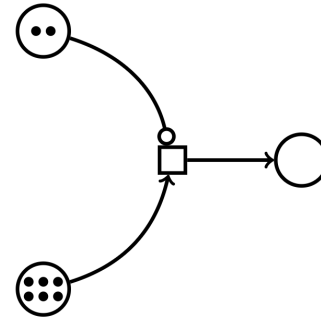


Fig. 14. Réseau de Petri avec arcs inhibiteurs.

Par exemple, malgré la présence de 6 jetons en amont de la transition que l'on voit sur la figure 14, elle ne peut pas être déclenchée parce qu'il y a 2 jetons dans la place en amont de l'arc inhibiteur.

Un théorème [5] affirme que pour toute fonction calculable, il existe un réseau de Petri à 2 (maximum) arcs inhibiteurs la calculant au sens vu précédemment (voir la section « Calcul par réseaux de Petri »). Voici-après des réseaux de Petri effectuant certaines opérations, et inédits jusqu'ici.

Multiplication

On voit sur la figure 15 un réseau de Petri à deux arcs inhibiteurs permettant d'effectuer une multiplication.

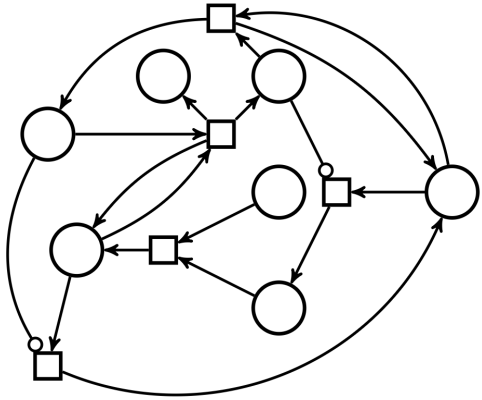


Fig. 15. Réseau de Petri pour multiplier.

Pour multiplier 3 par 2, on le marque comme on le voit sur la figure 16 (remarque : on peut aussi intervertir les places contenant 3 et 2 jetons, cela donne le même résultat, ce qui ne saute pas aux yeux).

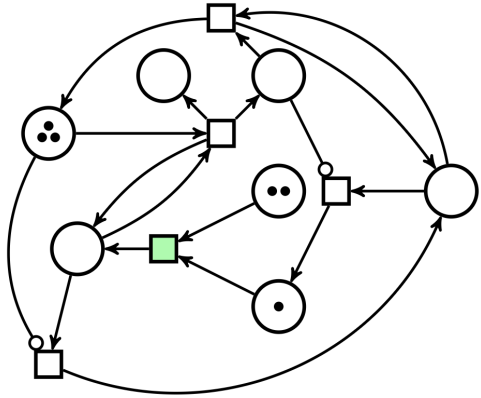


Fig. 16. Marquage initial pour multiplier 2 par 3.

Le jeton seul vers le bas est dans une place binaire : une telle place ne contient qu'un jeton maximum au cours du calcul. Ce jeton joue le rôle d'un *marqueur* permettant de contrôler les étapes de calcul (en s'arrangeant pour qu'une et une seule transition soit active à un instant donné[2]). Une fois le calcul terminé on a le produit dans la place finale (confer figure 17).

2. Les transitions inhibitrices servent à rendre synchrone un réseau de Petri, qui est par nature asynchrone.

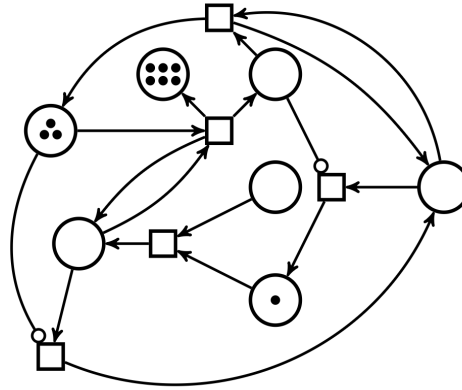


Fig. 17. Multiplication effectuée.

Division euclidienne

Le réseau de Petri de la figure 18 comporte 3 arcs inhibiteurs.

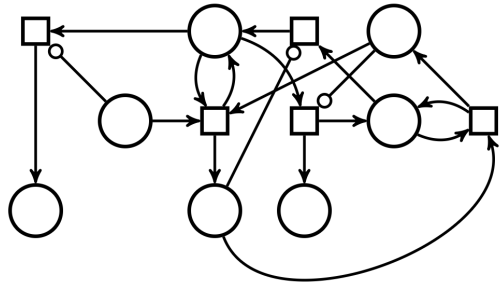


Fig. 18. Réseau de Petri avec trois arcs inhibiteurs.

Ce réseau effectue (par soustractions itérées) une division euclidienne. Par exemple pour diviser 5 par 2, on dispose un marqueur dans une place binaire, 5 jetons dans la place du dividende (cette place contiendra le reste à la fin du calcul) et 2 jetons dans la place du diviseur (ce réseau de Petri ne calcule rien si le diviseur est nul) ; voir la figure 19.

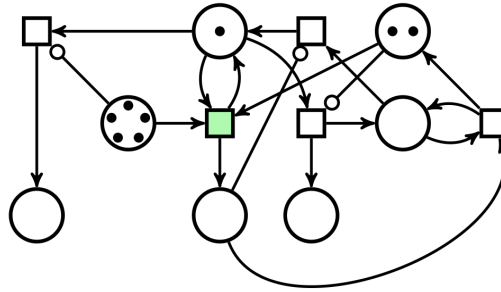


Fig. 19. Préparation de la division euclidienne.

Une fois que le réseau de Petri a terminé le calcul, le marqueur est dans la place binaire tout en bas à gauche, le reste 1 est dans la place suivante (en bas) et le quotient 2 est dans la place suivante (voir figure 20).

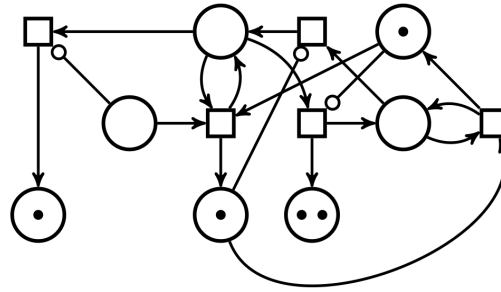


Fig. 20. Résultat de la division euclidienne.

Calcul de pgcd

Le réseau de Petri suivant matérialise l'algorithme d'Euclide (avec soustractions itérées) mais avec pas moins de 6 arcs inhibiteurs :

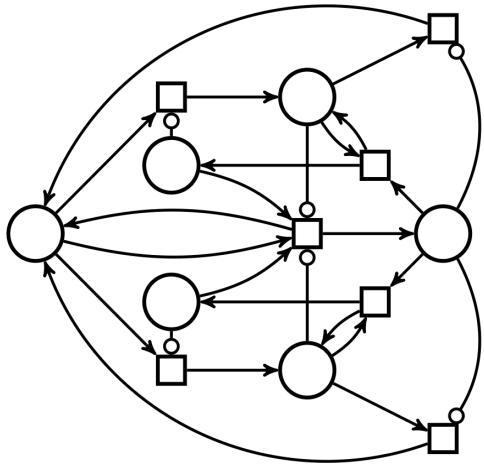


Fig. 21. Réseau de Petri permettant le calcul du pgcd.

Pour calculer le pgcd de 5 et 3, on place respectivement 5 jetons et 3 jetons dans les places prévues pour cela (celle qui sera non vide à la fin, contiendra alors le pgcd) et un marqueur dans la place binaire tout à gauche :

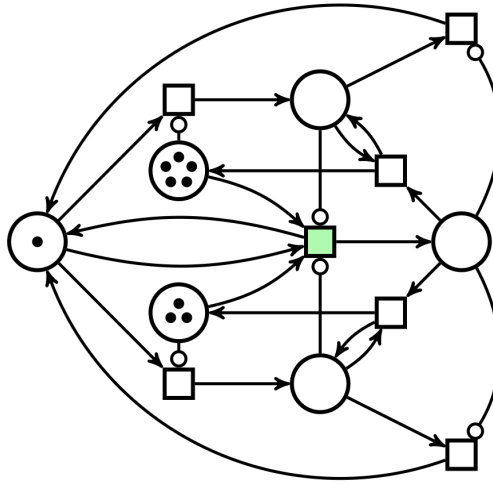


Fig. 22. Marquage pour préparer le calcul du pgcd.

Le calcul est terminé lorsque le marqueur est tout à gauche, et l'une des places suivantes est vide. L'autre contient le pgcd 1.

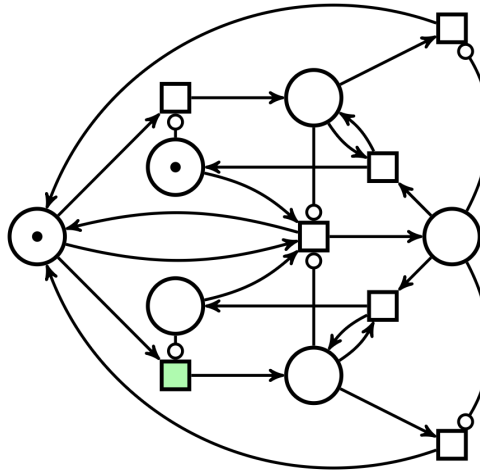


Fig. 23. Calcul du pgcd terminé.

Ce réseau de Petri a permis de faire calculer des pgcd en cycle 3, alors même que la notion n'est pas définie : il s'agit d'un jeu dont seule la règle est connue, et pas la motivation (vue en cycle 4). On voit là un exemple archétypal de *jeu sérieux* (dont la pratique permet d'apprendre sans même se rendre compte que l'on apprend). De plus, la pratique de ce réseau de Petri amène à des raccourcis comme le remplacement de 3 mouvements successifs de jetons, par un déplacement des 3 jetons d'un coup, une fois qu'on a compris (par la pratique) que cela revient au même.

On peut donc estimer que le réseau de Petri ci-dessus est une représentation graphique de l'algorithme d'Euclide avec une symétrie visible, qui prouve sans mot que $\mathrm{pgcd}(a,b) = \mathrm{pgcd}(b,a)$, et de même, les réseaux de Petri précédents représentent graphiquement les algorithmes des 4 opérations :

– addition par incrémentation d'une variable et décrémentation de l'autre, jusqu'à la nullité de celle-ci,
– soustraction par décrémentation simultanée des deux variables,
– multiplication par addition itérée,
– division euclidienne par soustraction itérée.

Les réseaux de Petri sont donc une activité débranchée sur les algorithmes (de calculs sur des vecteurs d'entiers) et permettent de découvrir par la manipulation, certaines propriétés de ces algorithmes, comme par exemple la symétrie du pgcd ou la notion d'invariant.

Références

[1] Alain Busser. 2019. Réseaux de Petri. Consulté à l'adresse `https://iremi.univ-reunion.fr/?p=627`.

[2] Alain Busser. 2020. *Jeux et Graphes*. Ellipses.

[3] Alain Busser. 2021. L'abaque de Neper, un artéfact quatre fois centenaire pour enseigner le binaire. Consulté à l'adresse `http://revue.sesamath.net/spip.php?article1432`.

[4] Carl Adam Petri. 1962. *Kommunikation mit automaten*. Dissertation, Schriften des IIM 2, Rheinisch-Westfälisches Institut für Instrumentelle Mathematik an der Universität Bonn.

[5] Dimitri Zaitsev et ZhuWu Li. 2017. On simulating Turing machines with inhibitor Petri nets. *IEEJ Transactions on Electrical and Electronic Engineering*, 13.

BI4people[1] – Le décisionnel pour tous

Jérôme Darmont
Université Lumière Lyon 2, ERIC

Enjeux et état de l'art

Les technologies de l'informatique décisionnelle (*Business Intelligence*, BI), telles que les entrepôts de données, qui permettent de collecter, stocker et structurer des données opérationnelles en données décisionnelles, ainsi que l'analyse en ligne (*On-Line Analytical Processing*, OLAP) qui aide à croiser plusieurs axes de données, sont des outils primordiaux dans l'aide à la décision dans de nombreux domaines comme la finance, mais aussi la santé ou l'enseignement supérieur [6, 8].

Les outils de l'informatique décisionnelle ont longtemps nécessité un investissement financier et humain très lourd. Toutefois, il existait au début du projet BI4people de nombreuses solutions de BI gratuites, qu'elles fussent propriétaires, libres ou infonuagiques [1]. Les logiciels propriétaires se focalisaient cependant sur les tableaux de bord et la visualisation, et avaient tous des fonctionnalités limitées, comme l'absence d'une intégration efficace de données depuis des sources disparates.

Bien que quelques logiciels libres proposassent des explorations OLAP, ils demeuraient techniquement hors de portée des petites entreprises, des associations, des chercheurs, des indépendants comme des journalistes ou des *makers*, et des citoyens actifs [2], que nous ciblions particulièrement dans le projet BI4people.

De plus, la tendance à déporter la BI dans le nuage avait rejoint la demande grandissante d'outils collaboratifs permettant aux usagers et usagères de croiser des données privées, publiques, ainsi que des *self data*,

1. https://eric.univ-lyon2.fr/bi4people/.

DOI:10.48556/SIF.1024.26.59

d'effectuer des analyses conjointes, d'annoter des figures ou des rapports et de communiquer via les réseaux sociaux [10]. Les réponses à l'époque à cette demande globale restaient en deçà des attentes en se limitant au partage en ligne de résultats d'analyse.

Objectifs

L'objectif de BI4people était de rendre accessible la puissance de l'analyse interactive OLAP à la plus large audience possible, en mettant en œuvre le processus d'entreposage de données en mode *software-as-a-service*, de l'intégration de données multisource, hétérogènes (typiquement sous la forme de tableaux issus de tableurs, de documents textuels ou semi-structurés, ou encore du Web) à une analyse OLAP et une visualisation très simples (figure 1).

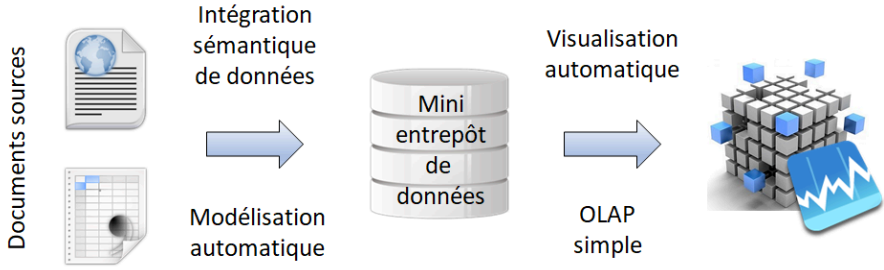


Fig. 1. Processus global.

Pour atteindre ce but, le service de BI devait inclure la *privacy by design*, être autonome, extrêmement simple, ergonomique et intelligible (jargon informatique ou BI interdit !). Dans ce contexte, les étapes classiques de l'entreposage de données s'appliquaient toujours, mais devaient être complètement automatisées [9, 12]. Au début du projet, BI4people était la première plateforme à atteindre complètement ce but, les projets similaires relevant plutôt de l'apprentissage automatique.

De plus, le prototype logiciel que nous proposions comme le délivrable principal du projet prenait en compte la confidentialité des données dans toutes les étapes, permettait des analyses collaboratives et était intelligible par ses usagers. Nous avions en effet insisté sur l'importance de l'appropriation des visualisations fournies par l'outil par les usagers, ce qui a impliqué une collaboration interdisciplinaire entre l'informatique (CNU 27) et les sciences de l'information et de la communication (CNU 71).

Enfin, il faut souligner que l'évaluation de nos prototypes par les usagers a été mise en œuvre tout au long du projet et pas uniquement à la fin.

Afin de mener tous ces travaux, nous avons constitué le consortium suivant :

– équipe de recherche de Lyon en sciences de l'information et de la communication (ELICO, appropriation des outils auprès des usagers) ;
– ERIC (informatique / BI collaborative, sécurité, Lyon) ;
– institut de recherche en informatique de Toulouse (IRIT, informatique / entrepôts de données) ;
– laboratoire d'informatique (LIFAT / visualisation, Tours) ;
– société TRIMANE (Saint-Germain-en-Laye, informatique / intégration, tests) ;

ainsi que des partenaires associés :

– *think-tank* FING, Marseille-Paris (jusqu'au 27/04/2022) ;
– TUBÀ Lyon (*living lab*) ;
– métropole de Lyon ;
– UrbaLyon.

Méthodes et approches

La méthodologie de projet que nous avons adoptée s'inspirait des méthodes agiles ou des *living labs*, mais sur des périodes plus longues, adaptées au rythme de la recherche. Concrètement, nous avions prévu de construire rapidement un premier prototype imparfait, puis de le perfectionner dans deux ou trois versions successives. L'idée était de confronter ce logiciel à différentes catégories d'usagers (des petites entreprises clientes du partenaire TRIMANE et des citoyens actifs recrutés par le TUBÀ), afin d'exploiter leurs retours et d'améliorer le prototype de manière incrémentale. Malheureusement, la pandémie de Covid-19 a bouleversé substantiellement le rythme du projet et nous n'avons pu avancer qu'une seule version, et avec moins de testeurs que prévu.

Néanmoins, nous avons conduit simultanément une étude d'utilité et d'appropriation du prototype par les usagers ainsi que leur évolution dans le temps, afin de mesurer si nos efforts allaient dans la bonne direction et amélioraient « l'expérience utilisateur ».

D'un point de vue opérationnel, le projet est subdivisé entre huit lots de travaux (*work packages*, WP), eux-mêmes scindés en sous-tâches, avec une prise en compte des risques potentiels et des solutions de repli (figure 2).

– WP1 — Coordination du projet ;
– WP2 — Automatisation de l'entreposage des données ;
– WP3 — Analyse collaborative des données ;
– WP4 — Visualisation et exploration des données ;

– WP5 — Protection des données ;
– WP6 — Validation et expériences ;
– WP7 — Évaluation de l'appropriation des usagers / usagères ;
– WP8 — Dissémination et exploitation.

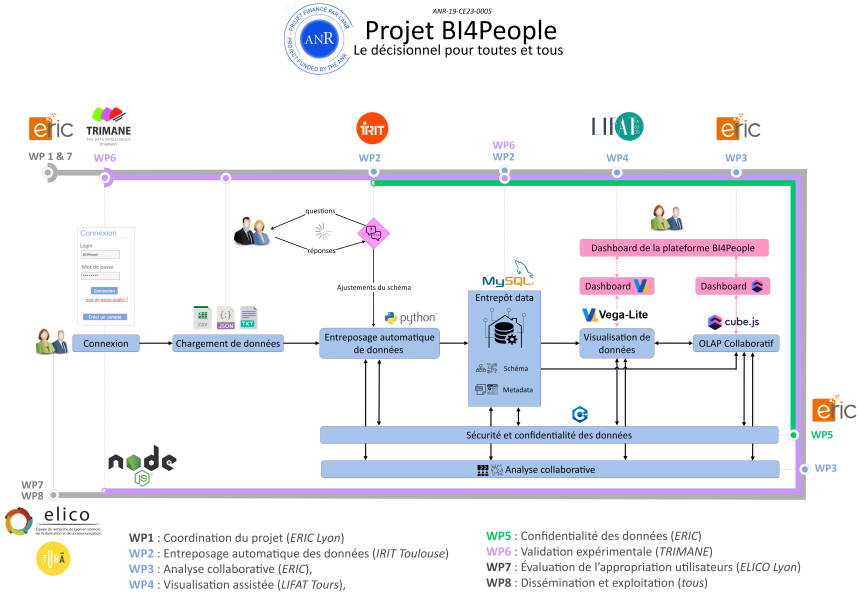


Fig. 2. Schéma global du consortium et des tâches du projet.

Nous avions prévu au sein du WP1 une approche intégrée avec tous les partenaires du projet, afin de prendre en compte pleinement les enjeux transversaux du projet, ainsi que de travailler sur la base d'un cas d'usage structurant. Toutefois, la temporalité des recrutements, les retards dus à la pandémie de Covid-19 et la dissolution de la FING (partenaire associé pourvoyeur de données) en 2022, nous ont forcé à nous adapter et à trouver des données de test plus diverses.

Résultats et faits marquants

Dans cette section, nous résumons les résultats principaux du projet BI4people, WP par WP.

WP1 — Coordination

Ce projet ambitieux traitait de différents volets informatiques, correspondant à des thématiques variées (entreposage de données, visualisation de données, sécurité, analyse collaborative) avec une collaboration avec les sciences de l'information et de la communication pour aborder la question de l'appropriation par les usagers. Il s'avère que chaque partie a pu aboutir à des résultats qui sont décrits dans ce qui suit en fonction des WP. Globalement, les objectifs scientifiques du projet ont été atteints, malgré quelques difficultés (cf. infra).

WP2 — Automatisation de l'entreposage

Nous avons conçu et implémenté une chaîne logicielle largement automatisée permettant de transformer des données tabulaires en entrepôt de données qualifiées [15, 16].

WP3 — Analyse collaborative

Nous avons conçu et implémenté deux prototypes permettant de :

1. enregistrer une démarche d'analyse pour un cas, diffusée ensuite aux autres personnes et qui enrichit la démarche de manière itérative [11] ;
2. utiliser un agent conversationnel dédié à l'aide à l'analyse [5].

WP4 — Visualisation

Nous avons créé un système de recommandation réalisant l'optimisation globale de tableaux de bords par un algorithme génétique [13, 14].

WP5 — Sécurité

Nous avons mené une étude des systèmes de chiffrement homomorphique, qui permettent de faire des calculs directement sur des données cryptées, protocoles de calcul sécurisés multipartites[2] pour sécuriser des cas d'usage [7].

WP6 — Validation

Nous avons mené une évaluation des parties développées. Chaque partie a été expérimentée et testée indépendamment. Les résultats obtenus sont encourageants pour une validation ultérieure plus complète.

2. Cela permet à plusieurs parties de faire un calcul sur leurs données respectives, sans que celles-ci ne puissent être divulguées aux autres parties.

WP7 – appropriation des usagers / usagères

Nous avons mené une étude des modalités d'appropriation de la dataviz à partir de ses prétentions communicationnelles, ses prédicats sémiotiques et les registres appréciatifs qui leurs sont associés par des usagers non-experts [3, 4].

WP8 – Dissémination

Le projet BI4people a produit 27 publications scientifiques, 2 thèses de doctorat, 5 mémoires de master, 3 vidéos, 1 poster ; 3 manifestations scientifiques ont été organisées, 4 présentations du projet ont été effectuées, 6 prototypes logiciels ont été réalisés et 1 jeu de données a été établi. Tous ces résultats sont libres de droits et déposés sur la plateforme HAL[3].

Perspectives générales

Globalement, aider l'analyse en permettant la production de résultats pour des usagers novices qui collaborent constitue une étape importante. Pour parfaire l'objectif d'accessibilité, il s'agit ensuite de soutenir l'étape d'interprétation des résultats. Ceci pourrait passer par une génération de l'analyse automatique ou semi-automatique. Une vigilance dans la recherche d'une telle solution est de présenter l'interprétation de telle ou telle partie d'une visualisation, qui n'est pas neutre, et qu'une interprétation plus globale est nécessaire. Une démarche collaborative trouverait également sa place pour parfaire l'interprétation des résultats et construire une connaissance commune sur les données traitées.

Finalement, l'interprétation de l'analyse rendrait possible d'aller plus loin en termes d'inclusion, notamment par rapport au handicap visuel. Les approches génératives de descriptions d'images sont un support important, mais cette perspective nécessiterait un travail pluridisciplinaire important pour prendre en compte les besoins des usagers.

Aider l'analyse en permettant la production de résultats pour des usagers novices qui collaborent constitue une étape importante. Pour parfaire l'objectif d'accessibilité, il s'agit ensuite de soutenir l'étape d'interprétation des résultats. Ceci pourrait passer par une génération de l'analyse (semi-)automatique. Une vigilance dans la recherche d'une telle solution est de présenter l'interprétation de telle ou telle partie d'une visualisation, qui n'est pas neutre, et qu'une interprétation plus globale est nécessaire. Une démarche collaborative trouverait également sa place pour parfaire l'interprétation des résultats et construire une connaissance commune sur les données traitées. Finalement, l'interprétation de l'analyse rendrait possible

3. https://hal.science/.

d'aller plus loin en termes d'inclusion, notamment par rapport au handicap visuel. Les approches génératives de descriptions d'images sont un support important, mais cette perspective nécessiterait un travail pluridisciplinaire important pour prendre en compte les besoins des usagers.

Enfin, nous souhaitons terminer sur le thème de l'interdisciplinarité, que nous avons voulue dès le début du projet. Sur la base de ce cheminement et de réflexions partagées, nous esquissons un protocole méthodologique qui se décline en trois phases.

1. Des enjeux et des temporalités différentes au sein de chaque WP.
2. Un sens consacré aux données différent au sein des sciences informatiques et des sciences de l'information et de la communication (SIC).
3. Quel sens attribuer aux données par les usagers ?
   – importance du « contexte d'usage » dans le processus d'appropriation ;
   – quelle place donner à l'usager dans la coconception des data-visualisation pour favoriser la « capacitation » ?

Références

[1] Top 30 open source and free business intelligence software. 2019. PAT RESEARCH, `https://www.predictiveanalyticstoday.com/open-source-freebusiness-intelligence-solutions/`.

[2] A. Abello, J. Darmont, L. Etcheverry, M. Golfarelli, J.-N. Mazon, F. Naumann, T.-B. Pedersen, S. Rizzi, J. Trujillo, P. Vassiliadis, and G. Vossen. 2013. Fusion Cubes: Towards Self-Service Business Intelligence. *International Journal of Data Warehousing and Mining* 9, 2: 66–88.

[3] T. Andry, J. Bonaccorsi, and F. Labarthe. 2022. Le décisionnel pour toutes et tous? In *Colloque Intersections du design - 3e édition - Le design dans la démocratie, Montréal (Québec), Canada*, 196–211.

[4] T. Andry, J. Bonaccorsi, and F. Labarthe. 2023. Le décisionnel pour toutes et tous? Retour sur les ambitions transformatrices d'un projet de recherche visant la démocratisation d'un outil d'analyse des données numériques. In *Intersections du design 2022 : Le design dans la démocratie, Montréal, Canada*, 212–227.

[5] O. Cherednichenko, F. Muhammad, J. Darmont, and C. Favre. 2023. A Reference Model for Collaborative Business Intelligence Virtual Assistants. In *6th International Conference on Computational Linguistics and Intelligent Systems (CoLInS 2023), Kharkiv, Ukraine* (CEUR), 114–125 vol. III.

[6] J. Darmont et P. Marcel. 2017. Entrepôts de données et OLAP, analyse et décision dans l'entreprise. CNRS Editions, Paris, 132–133.

[7] T.-V.-T. Doan, M.-L. Messai, G. Gavin, and J. Darmont. 2023. A Survey on Implementations of Homomorphic Encryption Schemes. *The Journal of Supercomputing* 79: 15098–15139.

[8] C. Favre, F. Bentayeb, O. Boussaïd, J. Darmont, G. Gavin, N. Harbi, N. Kabachi, and S. Loudcher. 2013. Les entrepôts de données pour les nuls… ou pas ! In *2e Atelier aIde à la Décision à tous les Etages (EGC/AIDE 13), Toulouse*.

[9] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter. 2018. Practical Automated Machine Learning for the AutoML Challenge 2018. In *ICML 2018 AutoML Workshop*.

[10] J. Kaufmann and P. Chamoni. 2014. Structuring Collaborative Business Intelligence: A Literature Review. In *In Proc. HICSS 2014*, 3738–3747.

[11] F. Muhammad and J. Darmont. 2023. An Ontology-based Collaborative Business Intelligence Framework. In *12th International Conference on Data Science, Technology and Applications (DATA 2023), Rome, Italy*, 480–487.

[12] L. De Raedt. 2016–2021. Synth: Synthesising Inductive Data Models.

[13] P. Soni, C. de Runz, F. Bouali, and G. Venturini. 2023. A genetic algorithm for automatic dashboard generation: first results. In *27th International Conference Information Visualisation (IV 2023), Tempere*, 77–82.

[14] P. Soni, C. de Runz, F. Bouali, and G. Venturini. 2024. A survey on Automatic Dashboard Recommendation Systems. *Visual Informatics*: Journal pre-proof.

[15] Y. Yang, F. Abdelhédi, J. Darmont, F. Ravat, and O. Teste. 2022. Automatic Machine Learning-based OLAP Measure Detection for Tabular Data. In *24th International Conference on Big Data Analytics and Knowledge Discovery (DaWaK 2022), Vienna, Austria* (Lecture Notes in Computer Science), 173–188.

[16] Y. Yang, J. Darmont, F. Ravat, and O. Teste. 2021. An Automatic Schema-Instance Approach for Merging Multidimensional Data Warehouses. In *25th International Database Engineering and Applications Symposium (IDEAS 2021), Montreal, Canada* (ICPS), 232–241.

Intégration de l'IA et de la biomécanique pour la prévention des blessures sportives

Estelle Delouche
Talan

Les blessures chez les sportifs de haut niveau représentent un enjeu majeur tant sur le plan physiologique que socio-économique. Elles impliquent des périodes de convalescence prolongées, une diminution du niveau de performance individuelle et collective, ainsi qu'un impact financier important pour les clubs professionnels.

La prévention des blessures pour les athlètes est donc devenue un axe de recherche important pour les clubs, mobilisant des équipes scientifiques pluridisciplinaires qui étudient à la fois des données physiologiques classiques (fréquence cardiaque « *Fc* » ou la VO_2 max) mais également les cycles d'activation musculaire en analysant les électromyogrammes (EMG).
Le projet de recherche présenté dans cet article résulte d'une collaboration entre l'ENSAM, le club de rugby du Stade français Paris, le centre de recherche et innovation Talan, et s'inscrit dans une démarche de prévention active des blessures, plus spécifiquement la rupture du ligament croisé antérieur (LCA), une lésion fréquente au niveau du genou.

Nous cherchons à déterminer si des schémas d'activation musculaire associés à un risque accru de rupture du ligament croisé antérieur sont identifiables dans les électromyogrammes.

DOI:10.48556/SIF.1024.26.67

Pourquoi étudier les EMG ?

Les signaux EMG enregistrent les phases de contraction et de relaxation musculaire, et offrent un aperçu de la coordination neuromusculaire. Ainsi nous pourrions déchiffrer les informations de ces signaux pour déterminer si une forme d'onde est représentative d'un mouvement à risque (un changement de direction dangereux pour le tendon par exemple). Cependant, comprendre les signaux EMG est une tâche complexe : ils sont également composés de bruit et d'artefacts qui couvrent les informations et compliquent les interprétations.

Dans ce contexte, l'électromyographie de surface (EMG) apparaît comme un outil de choix pour étudier les stratégies d'activation neuromusculaire susceptibles de prévenir ou, à l'inverse, de favoriser la survenue de lésions du LCA [8, 13, 7]. Introduite par Lemman et Ritchie en 1979, l'EMG de surface est une technique non invasive qui permet d'enregistrer, à l'aide d'électrodes placées directement sur la peau, l'activité électrique produite par les muscles lors de leur contraction.

L'analyse de ces signaux aide à la compréhension du système neuromusculaire en évaluant l'intensité et la coordination des activations des différents muscles. Dans le cadre de notre étude, un enregistrement simultané de plusieurs groupes musculaires a été réalisé, afin de cartographier la dynamique d'activation autour de l'articulation du genou. L'observation conjointe des signaux EMG des quadriceps et des ischio-jambiers permet ainsi d'identifier les contributeurs principaux au mouvement, et d'isoler les motifs d'activation potentiellement à risque, en particulier lors des phases critiques précédant un changement de direction [5].

Ainsi, le projet vise à développer une méthodologie d'analyse de ces signaux, dans le but d'identifier d'éventuels motifs d'activation musculaire associés à un risque accru de blessure.

Pourquoi utiliser l'intelligence artificielle ?

Les signaux EMG sont très riches en information sur l'activité musculaire, mais présentent également une forte complexité temporelle, une variabilité interindividuelle élevée et une présence importante de bruit. Les méthodes de statistiques classiques atteignent alors rapidement leurs limites. C'est dans ce contexte que l'intelligence artificielle, et plus particulièrement les techniques de *machine learning* et *deep learning*, apparaissent comme des outils prometteurs [15, 11, 9, 4].

Au-delà de la simple extraction de caractéristiques statistiques, l'intelligence artificielle offre un cadre pour modéliser les signaux EMG en tant que séries temporelles complexes. Les réseaux de neurones convolutifs (CNN)

permettent de détecter automatiquement des motifs locaux dans le temps, en apprenant des filtres capables de représenter les phases de contraction ou d'impact musculaire. Les architectures récurrentes, telles que les LSTM (*long short-term memory*) ou les GRU (*gated recurrent units*), facilitent quant à elles, la prise en compte des dépendances temporelles longues et des interactions dynamiques entre muscles. L'intégration de ces modèles dans une chaîne d'apprentissage supervisé permet de transformer l'analyse biomécanique en une tâche de reconnaissance de motifs temporels, où la performance est évaluée à l'aide d'indicateurs quantitatifs standard (taux de classification, précision, rappel, F1-score, ou encore AUC). Cette approche algorithmique vise à identifier des régularités latentes dans des signaux bruités et non stationnaires, tout en améliorant la robustesse et la possibilité de généraliser des modèles prédictifs.

Ainsi ces techniques sont particulièrement puissantes pour extraire automatiquement des motifs complexes à partir de données multivariées et bruitées et peuvent détecter des signaux faibles dans les signaux EMG [8, 12].

L'objectif de cette recherche est donc d'évaluer, à travers une approche combinant l'analyse de signaux EMG et des techniques d'intelligence artificielle, dans quelle mesure il est possible de discriminer des motifs d'activation musculaire potentiellement à risque, et de contribuer ainsi à une meilleure compréhension et anticipation des mécanismes opérant lors d'une LCA.

Comment surviennent les lésions du ligament croisé antérieur ?

Sur un plan anatomique, le ligament croisé antérieur s'insère sur la partie inférieure du fémur, traverse obliquement l'articulation du genou, puis vient se fixer sur la partie supérieure du tibia. Il forme avec le ligament croisé postérieur (LCP), une structure dite « croisée » qui permet de stabiliser le genou lors des mouvements de flexion, d'extension et de rotation. Principalement, il sert à protéger le genou pour restreindre la rotation ou translation antérieure et éviter la lésion.

Sa rupture correspond à une déchirure partielle ou totale du ligament, souvent à ses extrémités d'insertion, et résulte le plus souvent d'un traumatisme indirect [6, 2]. Dans le cas du rugby, les études ont montré que la majorité des ruptures du LCA surviennent sans contact direct avec d'autres joueurs, mais lors de mouvements brusques tels que des changements de direction rapides. Plus généralement cette rupture survient dans les 50 ms suivant l'impact du pied au sol [16].

Sur un plan biomécanique, deux facteurs ont été identifiés comme déterminants dans les ruptures du LCA :

– un déséquilibre du ratio de coactivation musculaire (notamment entre les quadriceps et les ischio-jambiers) ;
– un retard d'activation musculaire, ne permettant pas de stabiliser efficacement l'articulation lors de la phase critique d'appui.

Ces mécanismes conduisent à une surcharge articulaire non compensée, pouvant aboutir à une rupture ligamentaire [3], qui ne cicatrise généralement pas spontanément et requiert une intervention chirurgicale dans la majorité des cas.

Collecte de données et scénarios d'acquisition

Ainsi dans le cadre de notre projet nous souhaitons analyser l'activation musculaire des muscles qui protègent le genou afin de déterminer si un ou plusieurs enregistrent des phases dites à risque lors d'un changement de direction soudain. Pour cela des électrodes ont été placés sur deux groupes musculaires : les quadriceps et les ischio-jambiers. Un total de 5 muscles sont monitorés : le rectus femoris, le vastus lateralis, la vastus medialis, le biceps femoris et le semitendinosus (figure 1).

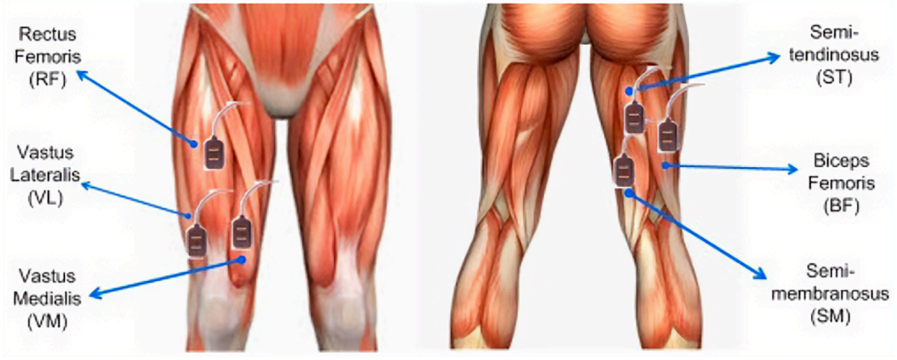


Fig. 1. Position des différents muscles observés dans notre étude. De *Bansal et al. [1].

Afin d'analyser les schémas d'activation musculaire dans des conditions réalistes, deux scénarios expérimentaux ont été conçus par les ingénieurs en biomécanique de l'ENSAM (figure 2). Le premier scénario, désigné dans la suite de l'article comme scénario *planifié*, consiste en une course linéaire de 15 mètres, au cours de laquelle le joueur de rugby récupère le ballon en mouvement avant d'effectuer un changement de direction prédéfini (à gauche

ou à droite), communiqué en amont de l'essai. Ce protocole vise à reproduire un déplacement maîtrisé dans un environnement contrôlé.

À l'inverse, le scénario *non-planifié*, également qualifié de scénario avec *incertitude*, vise à reproduire les conditions imprévisibles du jeu réel. Dans ce cas, le joueur ne connaît pas à l'avance la direction à prendre en fin de course. Pour simuler cette situation d'incertitude, un système de déclenchement visuel en temps réel a été mis en place. Lorsqu'un joueur franchit une paire de repères visuels (piliers jaunes) situés sur le parcours, un signal visuel instantané (flèche gauche ou droite) s'affiche sur un écran placé en face de lui, indiquant la direction à suivre immédiatement. Ce dispositif réduit le temps de prise d'information, contraignant le joueur à une prise de décision rapide, proche des exigences du jeu en situation réelle.

L'ensemble de ce protocole a été appliqué à un échantillon de 11 joueurs, chacun réalisant plusieurs essais dans les deux conditions. Au total, 1980 signaux EMG ont été enregistrés, fournissant une base de données pour l'application des algorithmes de *machine learning* et *deep learning*.

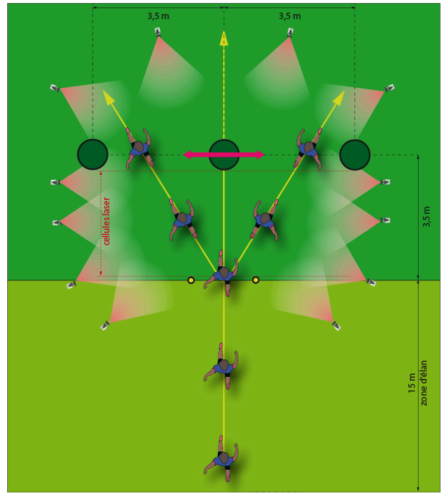


Fig. 2. Schéma du dispositif mis en place par l'ENSAM pour l'acquisition des données.

Pre-processing et processing des EMG

Les signaux électromyographiques (EMG) enregistrés nécessitent une phase de prétraitement et de traitement afin de garantir la fiabilité des analyses ultérieures. Dans un premier temps, le prétraitement vise à nettoyer et structurer les données brutes. Cela inclut l'identification et la suppression

des données aberrantes (figure 3). Le signal est ensuite centré (*demeaned*) pour éliminer toute composante continue résiduelle. Enfin, les portions de signal manquantes sont reconstruites par imputation temporelle locale, en les remplaçant par une fenêtre de même taille issue d'un segment antérieur immédiat du même signal, assurant ainsi la continuité et la cohérence temporelle et fréquentielle.
Concernant le traitement, un filtrage passe-bande entre 20 Hz et 450 Hz est appliqué afin de supprimer les artefacts de basse fréquence et les interférences haute fréquence [14]. Une normalisation du signal est ensuite envisagée, souvent par rapport à une mesure biomécanique de référence (par exemple, le pic de couple musculaire), afin de comparer les niveaux d'activation entre individus.

Dans la suite de cet article, nous présentons les résultats des algorithmes de *machine learning* sur des signaux EMG prétraités et filtrés mais où aucune normalisation n'a été appliquée.

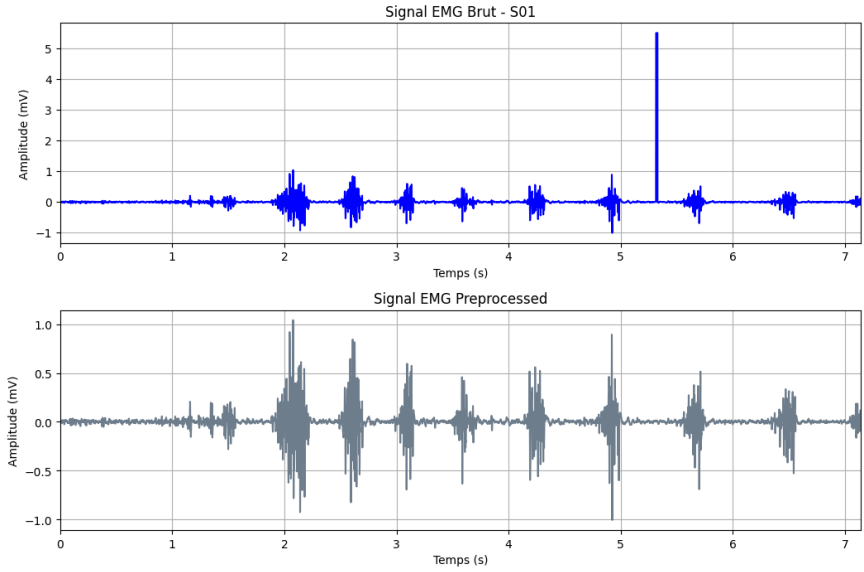


Fig. 3. De haut en bas : signal EMG brut et signal EMG après traitement (la donnée aberrante a été supprimée et le signal filtré).

Analyse des signaux EMG par apprentissage supervisé Extraction de caractéristiques et détection de variations

Nous avons exploré la capacité des techniques d'apprentissage supervisé à discriminer les signaux EMG issus de scénarios planifiés de ceux avec incertitude afin de déterminer si ces deux conditions induisent des motifs musculaires différenciables.

Pour cela, les signaux EMG ont été segmentés en fenêtres temporelles de 300 échantillons (soit environ 140 ms), avec un recouvrement de 50 %, afin de garantir une continuité dans l'analyse. Sur chacune de ces fenêtres, nous avons extrait un ensemble de caractéristiques tels que des descripteurs statistiques (moyenne, écart-type, kurtosis, *skewness*), des mesures d'énergie (entropies de Shannon et de Rényi), ainsi que des descripteurs de formes caractérisant la morphologie locale du signal.

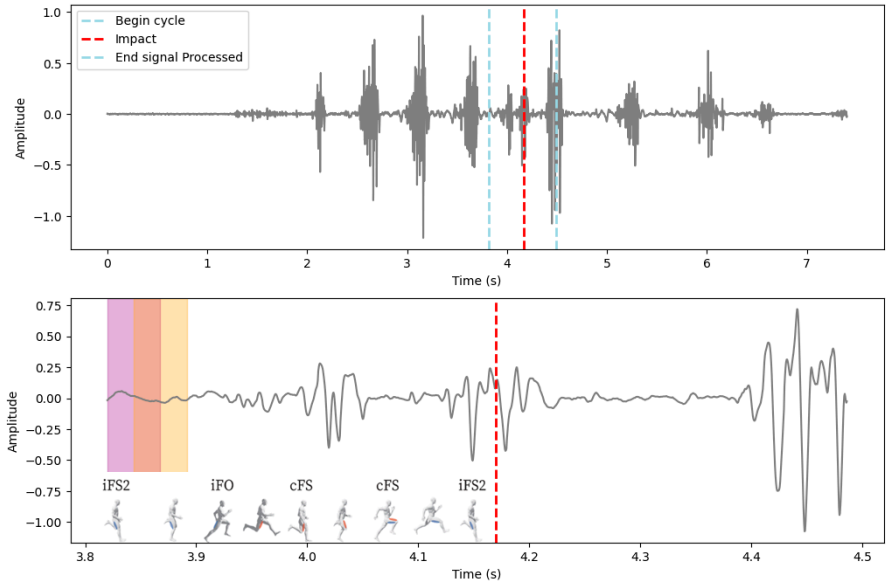


Fig. 4. Panel supérieur : exemple d'un signal EMG avec l'indication de la zone d'impact en lignes pointillées bleues. Panel inférieur : zoom sur la zone d'impact avec indication en bas à gauche du cycle de course (repris de [10]). Les rectangles colorés indiquent les fenêtres d'analyses du signal : 47 ms de longueur avec un overlap de 50 %.

Cependant, la dynamique globale étant fortement bruitée et difficile à interpréter, nous avons choisi de restreindre l'analyse à une zone d'intérêt physiologiquement critique (figure 4), centrée autour de l'impact initial du pied au sol précédant le changement de direction. Cette fenêtre temporelle de 0,7 seconde nous permet de centrer nos recherches de signaux autour de la zone le plus à risque. Dans cette configuration restreinte, les signaux ont été segmentés en fenêtres plus fines de 100 échantillons (soit 47 ms), ce qui correspond à l'échelle temporelle typique d'apparition du risque de lésion ligamentaire (environ 50 ms après l'impact final [16]).

L'ensemble des caractéristiques extraites dans cette zone a ensuite été utilisé pour alimenter des algorithmes de classification supervisé (*random forest, gradient boosting, decision tree*), dans le but de détecter l'existence éventuelle de sous-groupes de signaux différenciant les deux scénarios expérimentaux.

Les résultats indiquent des bonnes classifications de l'ordre de 72 % comme l'indique la matrice de confusion (figure 5).

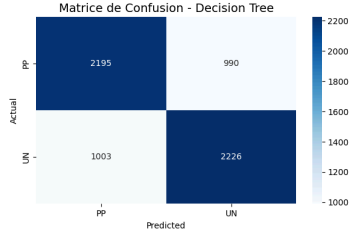


Fig. 5. Matrice de confusion calculée sur les résultats du *decision tree*. « PP » correspond aux signaux provenant d'un scénario planifié et « UN » fait référence au scénario avec incertitude.

Peut-on améliorer les prédictions ?

Compte tenu des performances limitées des algorithmes de classification supervisée, dont les taux de prédiction n'excèdent pas 72 %, nous avons exploré une approche complémentaire fondée sur la labellisation explicite des phases d'activation musculaire. L'objectif est de mieux caractériser la dynamique locale des signaux EMG, en distinguant les différentes phases physiologiques associées au mouvement, et en particulier la phase d'impact. Ainsi, nous avons attribué un label correspondant à 3 états possibles (figure 6) :

1. « repos » (non-contraction) ;
2. contraction musculaire active ;

3. impact, correspondant à la phase d'appui au sol précédant immédiatement un changement de direction.

La méthode de labellisation repose sur le calcul de la RMS (*root mean square*) du signal EMG. Un seuil empirique a été fixé pour discriminer les phases de repos (valeurs RMS inférieures au seuil) des phases de contraction (valeurs RMS supérieures). La zone d'impact, quant à elle, a été définie à partir d'annotations temporelles précises alignées sur les données cinématiques (moment du contact initial du pied au sol), et reçoit une étiquette dédiée. Cette stratégie permet de structurer les données EMG selon leur contenu fonctionnel et à extraire précisément les caractéristiques de chaque phase.
Ce travail de labellisation est actuellement toujours en cours mais nous pensons que les résultats pourront nous donner une meilleure compréhension de la dynamique des muscles et de mettre en évidence des aspects caractéristiques à chaque phase.

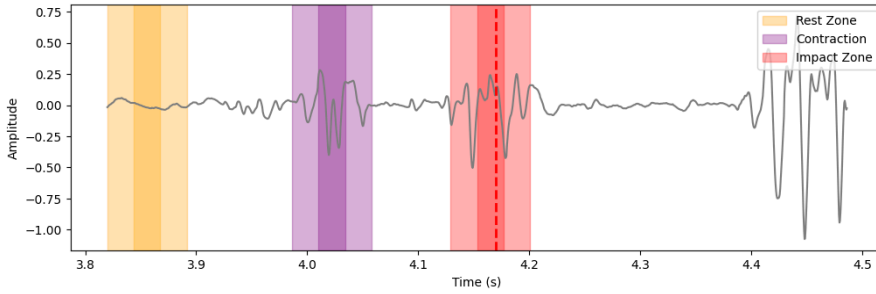


Fig. 6. Labellisation d'un signal selon le niveau de la RMS. En jaune, muscle considéré au repos, en violet en contraction et, finalement, rouge correspond à la zone d'impact.

Défis méthodologiques liés à l'analyse des signaux EMG

L'analyse et la modélisation des signaux électromyographiques se heurtent à plusieurs contraintes structurelles et physiologiques qui complexifient la détection de motifs généralisables :

1. un premier défi réside dans la variabilité temporelle de l'activation musculaire : même pour une tâche motrice identique, le moment d'activation des muscles peut différer d'un essai à l'autre, perturbant l'alignement temporel des signaux et rendant difficile la comparaison directe entre

cycles ; cette instabilité est exacerbée par les différences physiologiques interindividuelles, telles que la masse musculaire, le niveau de coordination neuromusculaire ou encore l'état de fatigue, qui modifient l'intensité et la dynamique des signaux enregistrés ;

2. le nombre de muscles étudiés : l'analyse des données EMG dans cette étude repose sur l'enregistrement simultané de cinq muscles par essai, soit un total de 1980 signaux ; cette granularité, bien qu'indispensable pour capturer la coordination musculaire autour du genou, engendre une complexité supplémentaire dans l'interprétation des résultats ; en effet, chaque muscle présente un motif d'activation propre, avec des temporalités et des rôles fonctionnels distincts dans le mouvement ; cela signifie que les données ne sont pas indépendantes, mais interconnectées dans une dynamique neuromusculaire globale, ce qui rend l'analyse plus complexe ;
3. enfin, la taille restreinte du jeu de données : l'étude repose sur un échantillon de 11 joueurs, tous issus de postes différents sur le terrain (pilier, talonneur, ailier, etc.), ce qui implique des profils physiologiques hétérogènes en lien avec les spécificités de leur rôle ; or, ces différences, en termes de morphologie ou de masse musculaire, ne sont pas prises en compte dans les analyses actuelles ; cela a pour effet de réduire la performance des algorithmes d'apprentissage et limite la possibilité d'identifier un motif d'activation commun ou caractéristique d'une situation à risque.

Conclusion

Ce projet de recherche, mené conjointement par l'ENSAM, le Stade Français Paris et le centre R&D de Talan, rassemble une équipe pluridisciplinaire de scientifiques afin d'identifier des motifs d'activation musculaire caractéristiques dans les signaux EMG.

Les premiers résultats montrent que l'extraction de caractéristiques permet déjà de distinguer les signaux issus de scénarios planifiés de ceux en « incertitude », avec un taux de bonne classification atteignant 72 %. Cependant, afin de mieux comprendre ces mécanismes, nous effectuons une labellisation des données afin d'extraire plus précisément les aspects caractéristiques des cycles musculaires et de la zone d'impact. À terme, nous souhaitons étendre cette analyse à des approches basées sur le *deep learning*, en mobilisant des architectures de réseaux neuronaux adaptées aux séries temporelles (telles que les CNN et les LSTM), afin de surmonter la forte variabilité intra et inter-individuelle des signatures d'activation musculaire.

Références

[1] H. Bansal, B. Chinagundi, P.S. Rana & N. Kumar. 2022. An ensemble machine learning technique for the detection of abnormalities in knee movement sustainability. *Sustainability*, *14*(20), 13464.

[2] N.A. Bates, N.D. Schilaty, A.J. Krych & T.E. Hewett. 2019. Variation in ACL and MCL strain before initial contact is dependent on injury risk level during simulated landings. *Orthopaedic journal of sports medicine*, *7*(11), 2325967119884906.

[3] T.F. Besier, D.G. Lloyd & T.R. Ackland. 2003. "Muscle Activation Strategies at the Knee during Running and Cutting Maneuvers:," *Medicine & Science in Sports & Exercise*, vol. 35, no. 1, pp. 119–127, DOI:10.1097/00005768-200301000-00019.

[4] L. Cervoni, R. Sleiman, D. Jacob & M. Roudesli. 2023. Explainable Artificial Intelligence in Response to the Failures of Musculoskeletal Disorder Rehabilitation. In International Workshop on Explainable Artificial Intelligence in Healthcare (pp. 14-24). Cham: Springer Nature Switzerland.

[5] S. Colby, A. Francisco, Y. Bing, D. Kirkendall, M. Finch & W. Garrett. 2000. Electromyographic and kinematic analysis of cutting maneuvers: implications for anterior cruciate ligament injury. *The American journal of sports medicine*, *28*(2), 234-240.

[6] F. Della Villa, M. Buckthorpe, A. Grassi, A. Nabiuzzi, F. Tosarelli, S. Zaffagnini & S. Della Villa. 2020. Systematic video analysis of ACL injuries in professional male football (soccer): injury mechanisms, situational patterns and biomechanics study on 134 consecutive cases. *British journal of sports medicine*, *54*(23), 1423-1432.

[7] C.J. De Luca. 2002. Surface electromyography: Detection and recording. *DelSys Incorporated*, *10*(2), 1-10.

[8] M.E. Hahn. 2007. "Feasibility of estimating isokinetic knee torque using a neural network model," *Journal of Biomechanics*, vol. 40, no. 5, pp. 1107–1114, DOI:10.1016/j.jbiomech.2006.04.014.

[9] D. Jacob, R. Tievant, L. Cervoni & M. Roudesli. 2023. Prédiction des blessures au Foot 5 à l'aide d'une méthode de machine learning. *Journal de Traumatologie du Sport*, *40*(4), 261-269.

[10] G. Kakehata, Y. Goto, S. Ido, and K. Kanosue. 2021. Timing of Rectus Femoris and Biceps Femoris Muscle Activities in Both Legs at Maximal Running Speed. *Med. Sci. Sports Exerc.*, Vol. 53, No. 3, pp. 643–652.

[11] B. Karlik. 2014. Machine learning algorithms for characterization of EMG signals. *International Journal of Information and Electronics Engineering*, *4*(3), 189.

[12] S.H. Park & S.P. Lee. 1998. EMG pattern recognition based on artificial intelligence techniques. *IEEE transactions on Rehabilitation Engineering*, *6*(4), 400-405.

[13] J. Sun, Y. Wang, J. Hou, G. Li, B. Sun & P. Lu. 2024. Deep Learning for Electromyographic Lower-Limb Motion Signal Classification Using Residual Learning. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *32*, 2078-2086. `https://doi.org/10.1109/TNSRE.2024.3403723`.

[14] J. Wang, L. Tang & J.E. Bronlund. 2013. Surface EMG signal amplification and filtering. *International Journal of Computer Applications*, *82*(1).

[15] J. Yousefi & A. Hamilton-Wright. 2014. Characterizing EMG data using machine-learning tools. *Computers in biology and medicine*, *51*, 1-13.

[16] M.K. Zebis, J. Bencke, L.L. Andersen, S. Døssing, T. Alkjær, S. P. Magnusson, M. Kjær & P. Aagaard. 2008. The effects of neuromuscular training on knee joint motor control during sidecutting in female elite soccer and handball players. *Clinical journal of sport medicine*, *18*(4), 329-337.

Paramétrisation de réseaux de régulation biologiques avec l'évolution artificielle et l'apprentissage par renforcement

Denis Pallez, Guillaume Grataloup
Jean-Paul Comet, Gilles Bernot
Université Côte d'Azur, CNRS, I3S

L'objectif de la modélisation des réseaux de régulation biologique (RRB[1]) est d'étudier et de comprendre les mécanismes qui permettent aux systèmes biologiques d'accomplir des fonctions essentielles, allant du métabolisme à l'adaptation aux perturbations environnementales. L'étude de la dynamique de ces systèmes ouvre de nouvelles perspectives avec des applications importantes en biologie fondamentale, de la biologie moléculaire à l'écologie, mais aussi en biologie synthétique, en bioproduction, en pharmacologie, en médecine ou en chronothérapie. Bien que de nombreux cadres aient été proposés pour modéliser les RRB, en particulier les réseaux de régulation génétique (GRN), les modèles obtenus sont souvent grossiers et ne tiennent pas compte des valeurs seuils régissant les transitions entre différents niveaux d'expression qualitative des gènes ou le temps passé dans un état donné.

Dans cet article, nous nous concentrons sur ces modèles grossiers, pour lesquels les valeurs précises des seuils sous-jacents sont inconnues. Nous formulons d'abord le problème clé de la modélisation, à savoir l'identification des paramètres, comme un problème d'optimisation sous contraintes. Cette étape d'identification est ensuite résolue à l'aide de métaheuristiques à base

1. En anglais : biological regulatory network — BRN.

DOI:10.48556/SIF.1024.26.79

de population, capables de gérer ces contraintes. Le modèle est ensuite affiné pour prendre en compte le temps passé dans des états qualitatifs, ce qui est rendu possible par le remplacement des paramètres qualitatifs par des paramètres continus appelés « célérités ». Ce nouveau problème d'identification est résolu à l'aide d'un problème d'optimisation sans contraintes, résolu avec succès à l'aide de métaheuristiques standard, surpassant ainsi les solveurs de contraintes continues (cCSP). Enfin, la recherche arborescente Monte Carlo (MCTS) a été améliorée dans le domaine continu afin de réduire le temps d'évaluation des solutions. Dans les deux derniers cas, les algorithmes sont adaptés pour trouver un ensemble diversifié de solutions. Ce travail montre que l'intelligence computationnelle permet de modéliser efficacement des réseaux de régulation à grain fin, et de proposer diverses solutions biologiquement plausibles.

1. Réseaux de régulation

Plusieurs cadres de modélisation ont été développés pour représenter les RRB, chacun offrant une perspective différente sur la dynamique du système. Le *cadre différentiel* utilise des équations différentielles ordinaires pour décrire l'évolution continue de l'expression des gènes dans le temps. Le *cadre stochastique*, quant à lui, prend en compte l'aspect aléatoire des processus biologiques, en modélisant les transitions entre états comme des phénomènes probabilistes. Enfin, le *cadre discret* propose une représentation abstraite en ne considérant qu'un nombre fini d'états, souvent binaire (présence ou absence d'une entité biologique) permettant une analyse qualitative du comportement du système. Toutes ces approches reposent en premier lieu sur la connaissance des activations (+) ou inhibitions (−) potentielles entre les entités biologiques : toutes ces interactions constituent un graphe *de régulation* ou *d'interaction* (figure 1, à gauche).

Malheureusement, le graphe d'interaction global n'est pas connu de manière exhaustive, et le nombre d'entités impliquées est trop grand pour qu'une approche globale puisse voir le jour. Ainsi, d'autres approches ont émergé : elles consistent à inférer un RRB à partir de nombreuses données biologiques expérimentales [13, 24, 9, 8]. La force de ces approches est de mettre au jour des interactions entre les entités biologiques qui n'étaient pas encore connues. Cependant ces méthodes ne permettent pas d'avoir les conditions précises dans lesquelles les activations ou inhibitions ont lieu. Les seuils de déclenchement des interactions constituent encore aujourd'hui une pierre d'achoppement pour la modélisation. Plus généralement, ce sont les *paramètres* qui régissent l'activité de chaque gène en fonction des autres gènes qui sont extrêmement difficiles à identifier car ils sont sensibles, d'une part, aux conditions expérimentales et, d'autre part, au niveau d'abstraction

de l'étude. Comme ces paramètres ne sont généralement pas directement mesurables par des expérimentations biologiques, ils nécessitent des raisonnements subtils et complexes pour déduire leur valeur du comportement global observé du système. Cette identification des paramètres constitue le goulot d'étranglement de la modélisation d'un RRB, tout particulièrement dans le cadre discret. C'est ce sur quoi travaille l'équipe de recherche en bio-informatique du laboratoire I3S[2] depuis de nombreuses années. L'une des voies envisagées pour cette identification des paramètres consiste à exploiter, non pas les données expérimentales brutes, mais des observations biologiques qualitatives issues de ces données brutes ainsi, bien sûr, que d'autres connaissances biologiques déjà établies par la communauté sur les systèmes biologiques étudiés. Ces observations sont formalisées en utilisant soit des logiques temporelles soit une version modifiée de la logique de Hoare [5] selon qu'elles représentent une connaissance sur la dynamique globale du système ou une suite d'observations issue d'une expérience. Cette formalisation permet de représenter l'ensemble des contraintes que le modèle du réseau de régulation biologique doit satisfaire pour qu'il soit en accord avec l'ensemble des connaissances disponibles. Toutefois, cette démarche est extrêmement fastidieuse et coûteuse en temps : la modélisation fine d'un RRB, qui nécessite la connaissance de son graphe d'interaction et des paramètres expliquant la dynamique, s'étale généralement sur plusieurs thèses en bio-informatique ou en biologie.

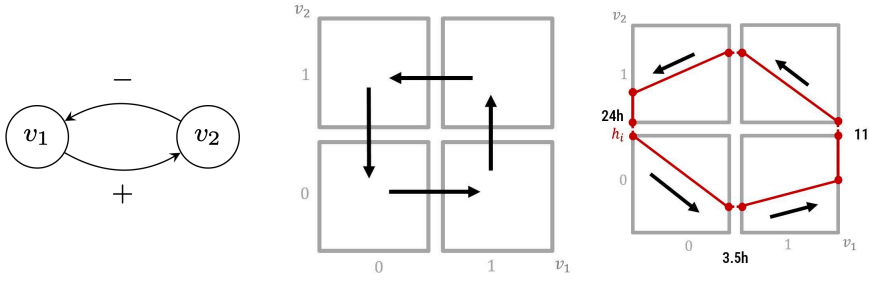


Fig. 1. À gauche : exemple d'un graphe d'interaction représentant le cycle circadien entre les gènes v_1 et v_2. Au centre : sa représentation discrète sachant que chaque gène possède 2 états discrets 0 ou 1. À droite : une dynamique possible dans le graphe hybride.

Le cadre discret de modélisation est souvent un point de départ intéressant car d'une part, les interactions sont souvent à seuil (sigmoïde) et, d'autre

2. Laboratoire d'informatique, signaux et systèmes de Sophia Antipolis, https://www.i3s.unice.fr.

part, l'accent est mis sur des comportements qualitatifs souvent plus facilement observables. Pour construire un modèle discret, il est avant tout nécessaire d'identifier le nombre d'états (discrets) de chacun des gènes du RRB. Ces états discrets sont le résultat d'abstraction de la concentration du produit des gènes : si on considère une activation d'un gène v_1 sur un gène v_2 , en dessous d'un certain seuil du produit du gène v_1, l'activation n'a pas lieu, et au dessus, elle a lieu. Cette notion de seuil est cruciale, et mène à une discrétisation de l'espace des états (cf. figure 1, au milieu). Vu le nombre de gènes et le nombre d'interactions impliqués, une approche manuelle n'est pas envisageable, et l'équipe de recherche tente de développer des approches permettant d'automatiser l'identification de ces seuils qui caractérisent les états discrets. L'idée de base est de partir des données brutes et de ramener le problème d'identification des seuils à un problème d'optimisation sous contraintes. Ce problème contraint est optimisé grâce à des métaheuristiques à base de population capables de gérer des contraintes exprimées sous forme d'inégalités, que nous expliquons dans la section 2.

Une fois les seuils identifiés, les états qualitatifs pertinents pour le système étudié sont évidents, et il est relativement facile d'expliquer la succession des événements, et par conséquent la succession des états discrets traversés au cours du temps. Toutefois, cela ne permet pas de déterminer le temps passé dans chacun des états, alors que la question temporelle traverse toute la biologie. C'est là qu'une extension *hybride* (RRBh) du cadre de modélisation discret prend tout son sens : le temps passé au sein de chaque état discret est déduit d'une trajectoire affine dont la direction et la vitesse sont représentées par un vecteur *célérité* (figure 1, à droite). Ainsi, les trajectoires du modèle sont des trajectoires linéaires par morceaux dans un espace multidimensionnel. Les paramètres de ces trajectoires, *i.e.* les vecteurs célérité, sont soumis à des contraintes : ils doivent permettre de représenter l'ensemble des observations qualitatives formalisées grâce à la logique de Hoare. Behaegel *et al.* [3] a logiquement utilisé des solveurs de satisfaction de contraintes continues (cCSP) pour échantillonner l'ensemble de toutes les paramétrisations compatibles avec les observations, mais les solveurs n'ont pas réussi à exhiber des solutions dès lors que le nombre de gènes du RRB dépassait trois. Pour exhiber tout de même des solutions, nous avons cherché à transformer ce problème de satisfaction de contraintes en un problème d'optimisation sans contrainte. Nous avons à nouveau adapté et comparé plusieurs métaheuristiques que nous détaillons dans la section 3.

Les paramètres recherchés sont les vecteurs célérité associés aux états traversés par la trajectoire, et cette trajectoire donne un ordre naturel dans l'ensemble des paramètres à rechercher. Cette particularité permet de considérer le problème d'identification des paramètres, dans un troisième temps, comme un enchaînement de décisions à prendre, état discret par état discret.

Pour cela, nous avons utilisé l'apprentissage par renforcement avec un algorithme de recherche arborescente Monte Carlo (MCTS) que nous détaillons dans la section 4.

Chacune des approches présentées dans ce travail a été testée sur un ou plusieurs systèmes biologiques distincts, notamment : un cycle de rétroaction négative à deux gènes (dont on attend un comportement périodique), un modèle abstrait du cycle circadien impliquant trois gènes, ainsi qu'un modèle du cycle cellulaire à cinq gènes. Enfin, nous exposons à la fin nos conclusions et les perspectives envisagées.

2. Optimisation sous contraintes

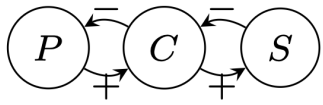


Fig. 2. Graphe d'interaction impliquant les gènes P, C et S.

À cette étape — et comme annoncé précédemment — nous supposons connu un graphe d'interaction (figure 2) ainsi que des données brutes qui représentent le niveau d'expression du produit des gènes (impliqués dans le graphe) au cours du temps (croix dans la figure 3, reliées en pointillés). L'objectif est de déterminer *automatiquement* les valeurs des seuils de chaque gène du RRB (traits horizontaux), ce qui se faisait jusque-là manuellement. Le seuil noté θ_P^{n+1} désigne une valeur scalaire représentant la concentration du produit du gène P qui sépare le $(n+1)$-ième état discret du n-ième. Dans un RRB, les seuils obéissent à des contraintes d'ordre, telles que $\theta_P^1 \leqslant \theta_P^2$, reflétant une progression croissante de la concentration du produit de P entre les états discrets 0 (en dessous des deux seuils), 1 (entre les deux seuils) et 2 (au dessus des deux seuils).

La connaissance du graphe d'interaction et des seuils de chaque gène ne permet pas, à elle seule, de définir la dynamique complète du réseau : les concentrations des produits des gènes évoluent dans le temps, provoquant potentiellement des transitions entre états discrets. Le passage d'un état à un autre d'un gène x est provoqué par des variations dans la concentration des produits des gènes qui précèdent x dans le graphe d'interaction. Toutefois, comme x peut avoir plusieurs prédécesseurs, il est essentiel de déterminer quels prédecesseurs ont une activité incitant x à changer d'état : cela peut être un activateur actif $(+)$ de x qui l'aide à s'exprimer suffisament pour que x franchisse son seuil, ou un inhibiteur actif $(-)$ dont la concentration diminue incitant x à s'exprimer, ou la combinaison de plusieurs prédécesseurs… Par la suite, x pourra faire évoluer à son tour d'autres gènes.

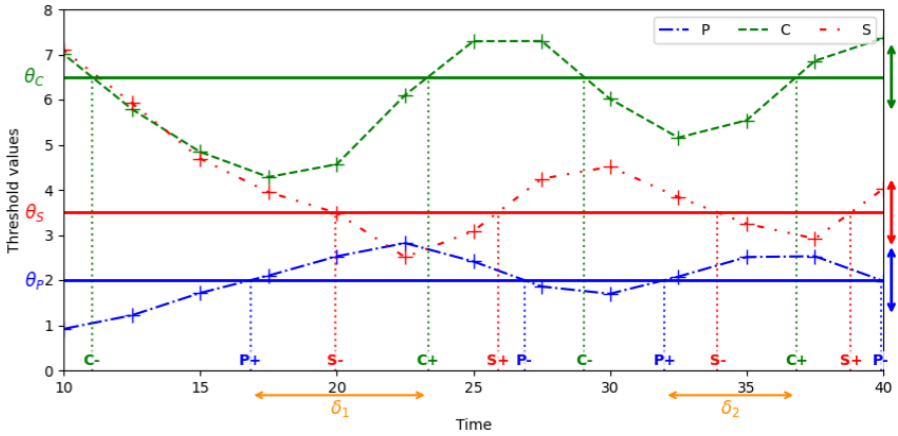


Fig. 3. Évolution de l'expression observée du produit des gènes du graphe de la figure 2 (croix) et les valeurs recherchées des seuils (traits pleins horizontaux) qui permettent de déduire une suite d'événements biologiques ($C-, P+, S-, \ldots$).

Ainsi, pour définir finement la dynamique d'un RRB, on a besoin de caractériser à la fois les seuils de chaque gène mais aussi ses ressources : une ressource est un prédécesseur qui aide le gène *cible* à s'exprimer, c'est à dire un activateur ($+$) qui dépasse son seuil d'activation, ou un inhibiteur ($-$) qui est en dessous du seuil d'inhibition. D'après le graphe d'interaction, il est possible de déduire :

– le nombre de seuils différents pour chaque gène x, sachant qu'il y a au maximum autant de seuils qu'il y a de régulations contrôlées par x ;
– le nombre de conditions différentes contrôlant le gène x : lorsque x a n prédécesseurs, il faut considérer toutes les situations distinctes où chaque prédécesseur peut être ou non une ressource ; ainsi, il y aura 2^n conditions possibles (les parties d'un ensemble à n éléments).

Pour chacune de ces situations, il est nécessaire de décrire le comportement du gène cible : vers quel état discret x va-t-il être attiré lorsqu'il est soumis à tel ensemble de ressources. Ces valeurs sont données par de nouveaux paramètres, notés $K_{x,\omega}$, qui donnent l'état discret vers lequel x est attiré lorsque ω est l'ensemble des ressources. En plus des contraintes d'ordre sur les seuils, il existe d'autres contraintes, dites de Snoussi [10], qui expriment le fait que la valeur de l'état discret vers lequel un gène est attiré, ne peut pas décroitre si ses ressources augmentent.

La figure 3 présuppose connues les valeurs des seuils des gènes du RRB (traits horizontaux). À l'aide des données brutes observées (croix), il est possible de savoir qu'un gène passe un seuil de manière croissante — la concentration de son produit a augmenté — ou décroissante — la concentration de son

produit a diminué. En suivant l'axe horizontal de cette figure représentant le temps, le premier événement pertinent est le fait que le produit du gène C (courbe supérieure de la figure) passe d'une valeur supérieure au seuil fixé de C (ligne horizontale) à une valeur inférieure. Cet événement est noté $C-$ au dessus de l'axe du temps de la figure. De manière similaire, le deuxième événement pertinent est $P+$ lorsque la courbe bleue en pointillé passe le seuil de P en montant. Le troisième est $S-$. La suite des événements est donc $C-, P+, S-, C+, S+, P-, C-, P+, S-, C+, S+, P-$. Notons qu'un changement des seuils (supposés connus) va entraîner un changement de cette suite d'événements.

Intéressons-nous à l'événement $C+$ qui apparaît deux fois dans la suite précédente. Cet événement peut être expliqué soit par l'augmentation de son activateur P, soit par la diminution de son inhibiteur S dans le graphe d'interaction (figure 2) et on constate par deux fois, dans la suite des événements, que $P+$ et $S-$ précèdent l'événement $C+$. On postule que si les seuils sont bien choisis alors le temps qui sépare l'apparition d'une ressource et l'augmentation de la cible est stable au cours du temps. Dans notre cas, il est donc utile d'évaluer la variabilité des délais (indiqués par des flèches orange en dessous de l'axe du temps sur la figure 3) entre les événements ressources ($P+$ ou $S-$) et l'événement conséquence $C+$. La proximité de ces différentes valeurs suggère une bonne adéquation des seuils associés. Il faut bien évidemment faire de même avec les autres gènes : on observe dans les données deux occurrences de $C-, P+$ en sachant que C est ressource de P. Ainsi, nous avons transformé l'identification des valeurs des seuils du RRB en un problème de minimisation de la variabilité des délais (variance) entre des événements biologiques en tenant compte de deux types de contraintes (d'ordre sur les seuils et de Snoussi).

Pour résoudre ce problème d'optimisation, nous avons considéré les 3 meilleures métaheuristiques à base de population issues de la compétition CEC'2020 [17] de la catégorie problèmes d'optimisation sous contraintes à objectif unique dans le monde réel[3]. Cette compétition a comparé 8 algorithmes d'optimisation gérant des contraintes sur 57 problèmes réels issus de différents domaines (chimie, mécanique, électronique… mais pas bio-informatique), et contenant entre 2 et 158 variables à optimiser, entre 0 et 148 contraintes d'égalité et entre 0 et 91 contraintes d'inégalité. Les algorithmes ont été classés en fonction de la somme pondérée des meilleurs résultats, de la médiane et de la moyenne obtenus sur les 57 problèmes ; une pondération plus importante étant accordée aux dimensions plus élevées. Les gagnants étaient *Self-adaptive spherical search* (SASS) [15], suivi de *Constrained optimization with Lévy flights differential evolution* (COLSHADE) [12] et de *Modified covariance matrix adaptation evolution strategy* (sCMAgES) [15].

3. https://github.com/P-N-Suganthan/2020-RW-Constrained-Optimisation.

Nous avons adapté ces algorithmes [11] à notre problème d'identification des seuils. La première étape de cette adaptation consiste à identifier les paramètres représentant une solution candidate Ψ manipulée par les algorithmes. Si $x_1, x_2, \ldots, x_n$ est l'ensemble des gènes du RRB et t_i le nombre de seuils de x_i, Ψ contient la valeur du premier seuil de chaque gène du RRB, $\theta^1_{x_i}$, à laquelle s'ajoute une valeur $\delta^1_{x_i}$ pour obtenir la valeur du seuil $\theta^2_{x_i} = \theta^1_{x_i} + \delta^1_{x_i}$, à laquelle s'ajoute la valeur $\delta^2_{x_i}$ pour obtenir la valeur du seuil suivant ($\theta^3_{x_i} = \theta^2_{x_i} + \delta^2_{x_i}$) et ainsi de suite jusqu'à $\theta^{t_i}_{x_i}$ et cela pour l'ensemble des gènes du RRB ($i \in [\![1, n]\!]$). Cela permet de tenir compte indirectement des contraintes de seuils introduites précédemment ($\theta^1_{x_i} \leqslant \theta^2_{x_i} \leqslant \theta^3_{x_i}$). L'algorithme sera en charge de vérifier uniquement les contraintes de Snoussi.

À ces valeurs, s'ajoutent également les paramètres K des ensembles de ressources $r^1_i = \varnothing, r^2_i, \ldots, r^{j_i}_i$ de x_i pour l'ensemble des gènes. Ainsi, $\Psi = [\theta^1_{x_i}; \delta^1_{x_i}; \ldots; \delta^{t_i-1}_{x_i}; K_{x_i,r^1_i}; \ldots; K_{x_i,r^{j_i}_{x_i}}]_{i=1,\ldots,n}$. Il est important de noter que les valeurs $\theta^j_{x_i}$ et $\delta^{t_i}_{x_i}$ sont réelles alors que les $K_{x_i,r^{j_i}_i}$ sont entières. Pour le RRB de la figure 2 qui considère que les seuils des régulations $C \to P$ et $C \to S$ sont identiques, Ψ contiendra 3 réels (les seuils des 3 gènes) et 8 entiers, $\$K_{P,\varnothing}$, $K_{P,C}$, $K_{C,\varnothing}$, $K_{C,P}$, $K_{C,S}$, $K_{C,PS}$, $K_{S,\varnothing}$ et $K_{S,C}$, qui doivent respecter 6 contraintes de Snoussi

$$K_{p,\varnothing} \leqslant K_{P,C}, \quad K_{C,\varnothing} \leqslant K_{C,P}, \quad K_{C,\varnothing} \leqslant K_{C,S},$$
$$K_{C,P} \leqslant K_{C,PS}, \quad K_{C,S} \leqslant K_{C,PS}, \quad K_{S,\varnothing} \leqslant K_{S,C}.$$

La deuxième étape consiste à définir la fonction d'évaluation d'une solution candidate et nous utiliserons la technique expliquée précédemment qui consiste à minimiser la variance des délais des différents passages de seuils, et ce, sur l'ensemble des gènes.

Nous avons exécuté chaque algorithme 50 fois avec à chaque fois, la même population initiale pour chacun des algorithmes, composée de 100 solutions candidates choisies aléatoirement. Le nombre de mesures expérimentales (croix) pour chaque gène est de 30 équitablement réparties dans le temps. À chaque algorithme est alloué un budget maximum de 10 000 évaluations de la fonction à minimiser (*number of function evalutions*, NFE). L'évolution moyenne des 50 exécutions de la qualité des solutions au cours du temps (NFE) est représentée sur la figure 4 pour chaque algorithme. On constate que SASS converge rapidement vers une solution optimale car la variance des délais devient nulle alors que sCMAgES converge vers un optimal local. COLSHADE semble converger mais beaucoup plus lentement que SASS qui obtient un optimum global après 7 000 évaluations. Sur cette figure, on peut également constater que les populations de COLSHADE et sCMAgES conservent une certaine diversité alors que c'est beaucoup moins le cas pour SASS.

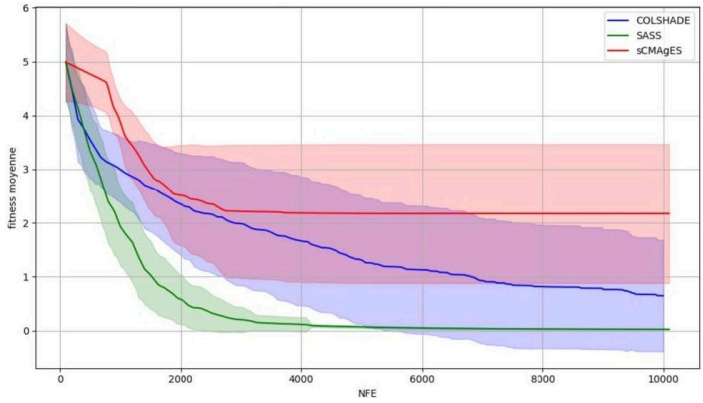


Fig. 4. Courbes de convergence des trois métaheuristiques COLSHADE, SASS et sCMAgES.

La figure 5 présente les fonctions de répartition (*cumulative distribution function*) qui décrivent, pour chaque algorithme, la probabilité (en ordonnée) de trouver une solution ayant une qualité égale ou inférieure à une valeur donnée (en abscisse).

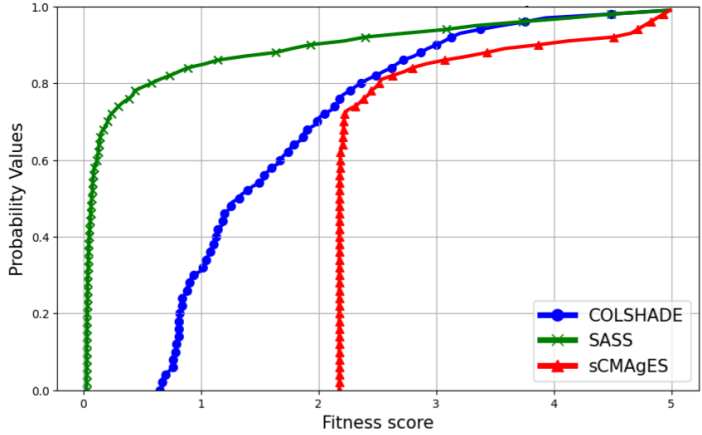


Fig. 5. Fontions de répartition des trois métaheuristiques COLSHADE, SASS et sCMAgES.

Ainsi, on constate que sur 50 exécutions, SASS a une probabilité d'environ 60 % de trouver une très bonne solution d'une qualité inférieure à 0.2 alors que les deux autres algorithmes n'y arrivent jamais. Non seulement, cela confirme les résultats de la compétition CEC'2020 [17] qui avait été gagnée par SASS mais surtout cela confirme qu'il est possible de déterminer automatiquement des seuils à partir de données brutes.

Par la suite, nous devons étudier le passage à l'échelle de notre approche, en l'appliquant à des RRB de plus grande taille, tout en déterminant le nombre minimal de mesures expérimentales nécessaires pour identifier les seuils. On pourra noter que nous avons utilisé des algorithmes d'optimisation conçus pour des variables continues, alors que les paramètres manipulés sont mixtes (entiers et réels). Cela implique que les algorithmes doivent explorer un espace de recherche beaucoup plus vaste que dans le cas d'un problème purement continu, ce qui peut expliquer la faible probabilité d'obtenir une solution satisfaisante. À l'avenir, nous envisageons de recourir à des algorithmes d'optimisation mixte, capables de gérer efficacement différents types de variables, dans l'espoir d'améliorer la convergence et d'augmenter la probabilité d'atteindre une solution optimale.

3. Optimisation sans contraintes

Dans ce nouveau travail, nous considérons les seuils des gènes comme identifiés mais nous passons du cadre de modélisation discret (figure 1, au centre) au cadre de modélisation hybride (figure 1, à droite) qui ajoute au cadre discret le temps passé dans chacun des états. Cela conduit à la recherche de trajectoires linéaires par morceaux dans un espace multidimensionnel. Ces trajectoires doivent respecter une séquence d'événements biologiques observés, qui a été précédemment et manuellement interprétée par les biologistes comme un ensemble de contraintes basées sur la logique de Hoare [5]. Les trajectoires sont caractérisées par une collection de célérités $\{C_{v,\eta}\}$ indexées par un des gènes v et un état discret η (correspondant à la vitesse d'évolution de la concentration du produit du gène v dans l'état η) et par un point de départ h_i. Ainsi, la représentation d'une solution candidate de la figure 1 (à droite) est $\Psi = [h_i; C_{v_1,(0,0)};\ C_{v_2,(0,0)};\ C_{v_1,(1,0)};\ C_{v_2,(1,0)};\ C_{v_1,(1,1)}; C_{v_2,(1,1)}; C_{v_1,(0,1)}; C_{v_2,(0,1)}]$ dont toutes les valeurs sont réelles.

Pour identifier les paramètres des trajectoires qui respectent les contraintes biologiques, nous avons tout d'abord transformé le problème de satisfaction de contraintes en un problème d'optimisation sans contrainte (*free optimisation problem*) en définissant une fonction de pénalité à minimiser qui calcule dans quelle mesure une solution candidate satisfait les contraintes [20].

Nous avons ensuite comparé plusieurs métaheuristiques telles que *Differential evolution* (DE), *Particle swarm optimisation* (PSO), *Covariance matrix adaptation evolution strategies* (CMA-ES), *Genetic algorithm* (GA), et *Random optimisation* (RO) qui sert de référence, et nous avons montré qu'elles sont capables d'identifier une solution équivalente à celles trouvées par les solveurs cCSP. Les résultats sous forme de fonction de répartition des différentes métaheuristiques sont présentées dans la figure 6. CMA-ES obtient

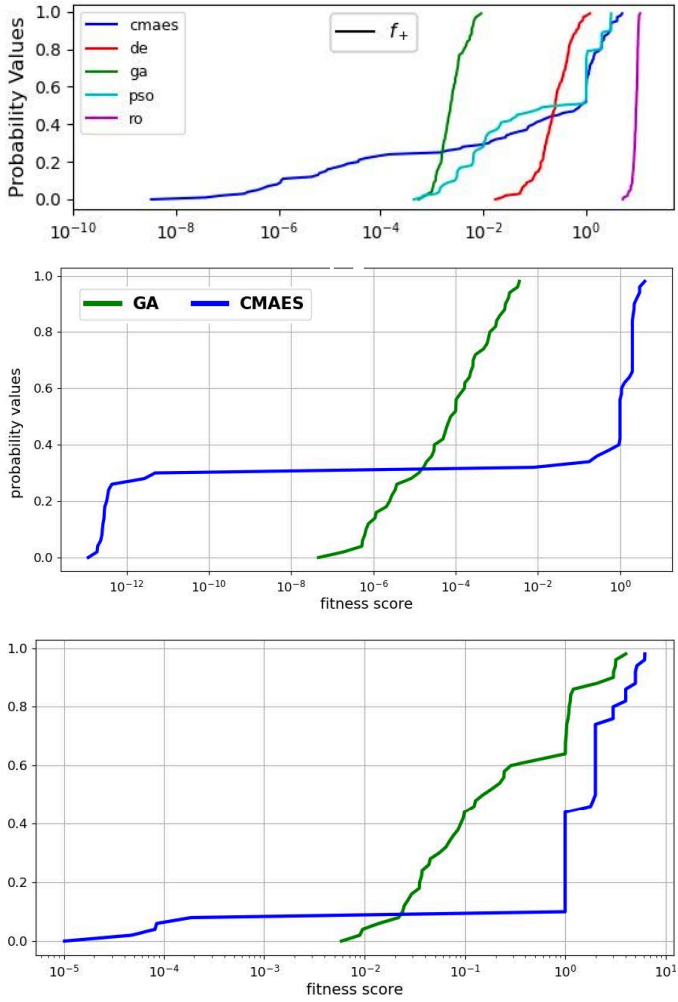


Fig. 6. Fonctions de répartition des différentes métaheuristiques sur un RRB à 2 gènes (en haut), 3 gènes (au centre) et 5 gènes (en bas), pour identifier les paramètres des célérités.

d'excellents résultats (valeur d'évaluation égale à 10^{-8} pour 2 gènes — voir figure 6, en haut — avec un optimum global à 0) mais avec une très faible probabilité (proche de 0) alors que GA obtient des résultats très intéressants

d'un point de vue biologique (une précision de 10^{-2} est suffisante) mais avec une très forte probabilité. Ces métaheuristiques ont même été capables d'identifier une solution pour des RRB de plus grande taille (5 gènes, cf. figure 6, en bas) pour lesquels les cCSP échouaient.

Multiples solutions ?

Les métaheuristiques à base de population considèrent généralement une structure panmictique (qui assure à chaque individu une probabilité identique d'interagir avec tous les autres membres de la population). L'inconvénient de ces approches est une diminution de la diversité de la population qui à terme converge vers un seul bassin d'attraction de l'espace de recherche alors que la connaissance de la diversité des solutions est importante pour le biologiste. À l'inverse, les algorithmes génétiques *cellulaires* [2], inspirés de la modélisation de systèmes complexes à l'aide d'automates cellulaires, impose une structuration de la population suivant une certaine topologie obligeant les échantillons à n'interagir (se reproduire) qu'avec leurs voisins proches dans la structure (figure 7).

Cette restriction d'interaction et le chevauchement de voisinage aident à l'exploration de l'espace de recherche tout en réduisant l'influence qu'un échantillon peut avoir sur l'ensemble de la population. L'intérêt de ce modèle d'évolution est qu'il permet de maintenir une certaine diversité dans la population favorisant ainsi une optimisation multimodale [14] au sein d'une même population.

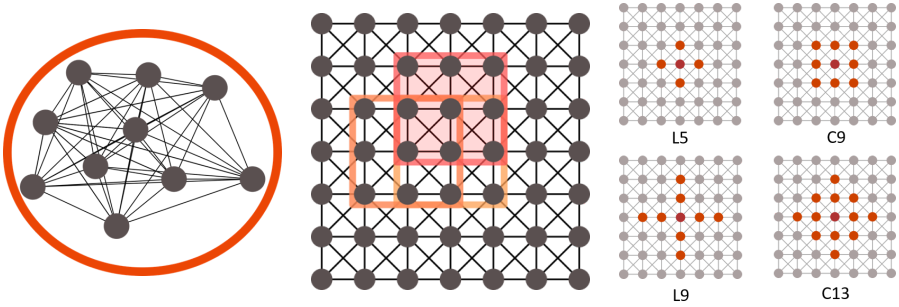


Fig. 7. Population panmictique (à gauche) et population possédant une certaine structure utilisée dans les algorithmes cellulaires (au centre) avec différents voisinages possibles (à droite). Un point correspond à une solution et une arête à une interaction possible.

Dans [19], afin d'avoir une situation comparable produite par les CSP (qui caractérisent une infinité de solutions), nous avons adapté ce schéma d'optimisation à notre problème pour obtenir plusieurs trajectoires optimales pour des RRBh de différentes tailles (2, 3 et 5 gènes). Pour cela, nous avons testé plusieurs structures de voisinage différentes (L5, L9, L29, L41, L13, C9, L pour linéaire et C pour compact, figure 7, à droite). Ces approches ont été comparées à l'algorithme *Repelling subpopulations in covariance matrix self-adaptation evolution strategy II* (RS-CMSA-ESII) [1], vainqueur de la compétition CEC'2020 [17] de la catégorie « techniques de niche pour l'optimisation multimodale[4] sur une dizaine de problèmes de tests [18]. Il a la particularité d'être basé sur l'algorithme CMA-ES auquel nous nous sommes comparés [20] et qui est capable d'obtenir d'excellents résultats mais avec une très faible probabilité. RS-CMSA-ESII est basé sur un schéma de coévolution visant à trouver des optima distincts (un par population) et utilise des points tabous de l'espace dans chaque sous-population pour construire de nouveaux échantillons qui sont éloignés des points tabous en se basant sur la distance de Mahalanobis.

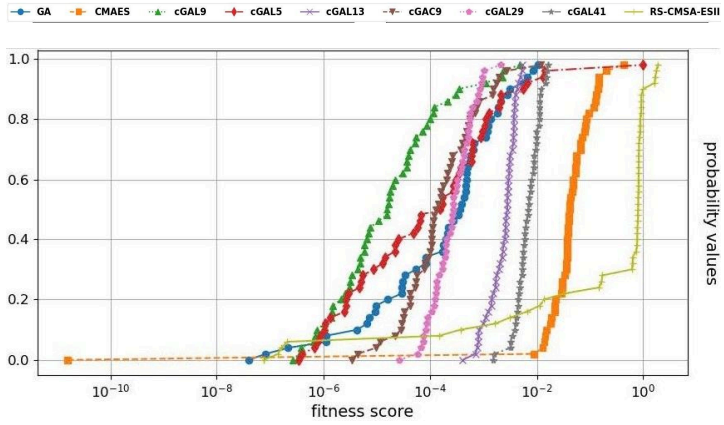


Fig. 8. Fonctions de répartition de plusieurs métaheuristiques avec un RRBh à 2 gènes moyennées sur 50 exécutions.

Nos expérimentations ont été répétées 50 fois avec un budget de 100 000 évaluations pour des RRBh à 2 gènes (figure 8) et 3 gènes (figure 9) et 200 000 évaluations pour un RRBh à 5 gènes (figure 10).

4. https://github.com/mikeagn/CEC2013/tree/master, https://www.epitropakis.co.uk/ieee-mmo.

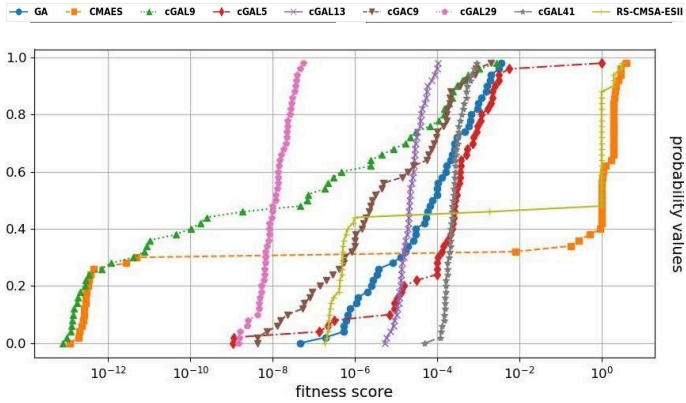


Fig. 9. Fonctions de répartition de plusieurs métaheuristiques avec un RRBh à 3 gènes moyennées sur 50 exécutions.

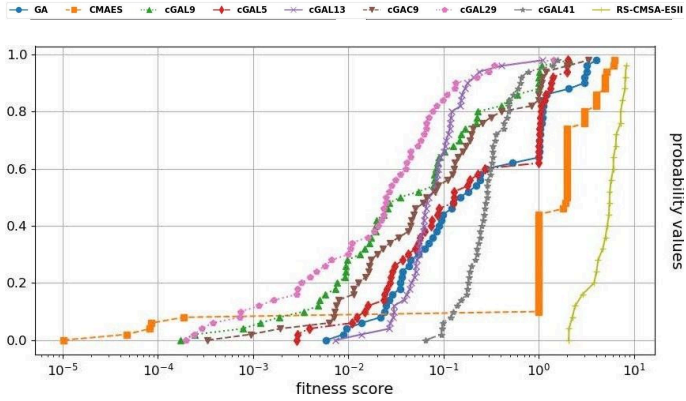


Fig. 10. Fonctions de répartition de plusieurs métaheuristiques avec un RRBh à 5 gènes moyennées sur 50 exécutions.

On peut observer que, comme prévu, les métaheuristiques panmictiques (GA, CMA-ES) sont moins performantes que l'algorithme génétique cellulaire (cGA) dans tous les cas car elles convergent plus rapidement vers un optimum local alors que cGA converge plus lentement mais en maintenant plus de diversité dans la population. Les résultats se confirment pour CMA-ES qui reste la technique qui obtient les meilleurs résultats mais avec une trop faible probabilité. La partie surprenante est que RS-CMSA-ESII obtient de moins bons résultats que CMA-ES et qu'il est souvent à la traîne pour ce qui est de trouver des solutions de qualité.

Nous avons mené une campagne de tests statistiques qui démontrent que sur l'ensemble des trois problèmes (2 à 5 gènes), cGAL9 et cGAL29 sont plus compétitifs que les autres techniques pour trouver plus d'optima avec des échantillons de meilleure qualité. Il est surprenant que RS-CMSA-ESII soit aussi contre-performant sur sa capacité à obtenir une grande diversité d'échantillons alors qu'il a gagné la compétition CEC'2020. En fait, cela est dû au fait que RS-CMSA-ESII ne garde au mieux qu'une seule solution par bassin d'attraction afin de pouvoir conserver d'autres solutions dans d'autres bassins. Or, notre problème d'optimisation est très particulier puisqu'il peut contenir une infinité de solutions optimales dans le même bassin d'attraction (plateau optimal, figure 12).

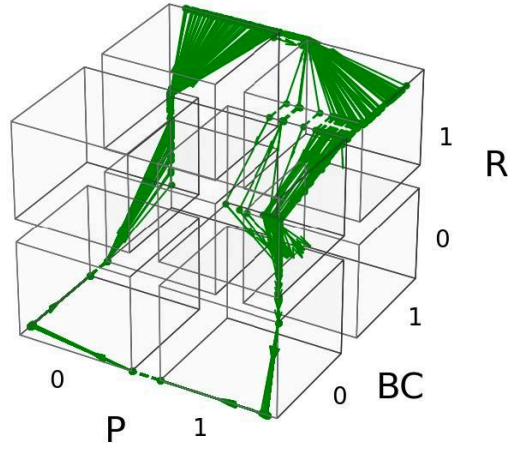
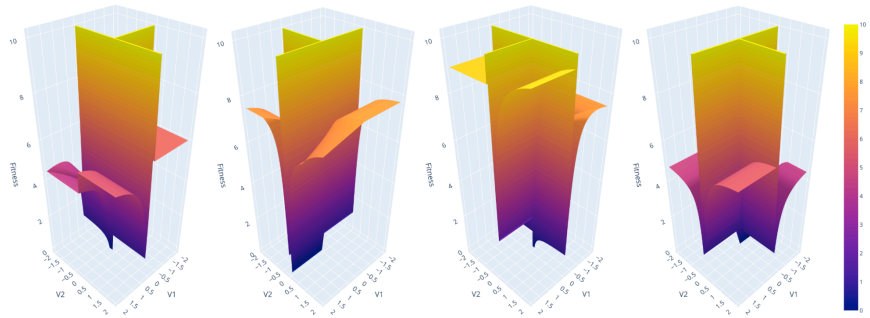


Fig. 11. Trajectoires diversifiées obtenues en une seule exécution avec cGAL9 sur un RRBh à 3 gènes.

Sur la figure 11, on peut observer les trajectoires qui valident les observations biologiques obtenues en une seule exécution de cGAL9 sur un RRBh à 3 gènes.

L'analyse des fonctions de test des compétitions sur les problèmes multimodaux [18] montre qu'aucune ne correspond aux caractéristiques de notre problème, soulignant ainsi les limites des fonctions de test actuellement utilisées dans la communauté. Ce problème de modélisation de réseaux de régulation a été proposé à la communauté scientifique dans le cadre de notre implication au sein du réseau européen COST *Action randomised optimisation algorithms research network*[5] et du groupe de travail consacré à la modélisation de problèmes.

5. https://roar-net.eu.



(a) (b) (c) (d)

Fig. 12. Visualisation de l'espace de recherche des trajectoires du RRBh à 2 gènes décomposé état discret par état discret qui fait apparaître en violet l'infinité de solutions : $(v_0 = 0, v_1 = 0)$ en (a), $(1, 0)$ (b), $(1, 1)$ (c), $(0, 1)$ (d).

4. Problème de décision séquentielle

Dans l'approche précédemment décrite, chaque solution candidate représente l'ensemble des célérités définissant une trajectoire à simuler. Si cette trajectoire conduit prématurément le système vers un état stationnaire, la simulation initiale ne permet pas d'en évaluer la suite éventuelle, nécessitant ainsi une relance. Ce processus conduit à l'évaluation de solutions non viables, augmentant inutilement le temps de calcul et pénalisant les performances de l'algorithme.

Dans ce nouveau travail, nous adoptons une stratégie alternative complètement différente qui consiste à identifier les célérités de manière séquentielle suivant l'ordre chronologique comme un processus de décision markovien (MDP) à états discrets mais dont les transitions entre les états sont continues et multidimensionnelles (les célérités). L'approche vise à sélectionner, pour le premier état discret du RRBh, une action (célérité) parmi l'ensemble des possibles, de manière à satisfaire les contraintes propres à cet état. Ce choix conditionne le point d'entrée dans l'état suivant, où le processus est reconduit. La succession de ces décisions permet de calculer une récompense cumulative, que l'on cherche à maximiser. Plus grand est le nombre d'actions identifiées respectant les contraintes biologiques, meilleure est la solution. L'intérêt est de construire et évaluer progressivement la solution, célérité après célérité.

Pour cela, nous avons adopté une méthode d'apprentissage par renforcement telle que la recherche arborescente Monte-Carlo (*Monte-Carlo tree search*, MCTS, figure 13) mise en lumière quand cette technique, combinée à de l'apprentissage profond, a battu le champion du monde de Go [25]. Le

principe consiste à construire progressivement un arbre de recherche dont les nœuds représentent les états et les arêtes les actions entreprises pour passer d'un état à un autre ; la récompense est obtenue en appliquant des simulations (figure 13.d). Les articles [6] et [26] présentent en détail la technique ainsi que de nombreuses variantes.

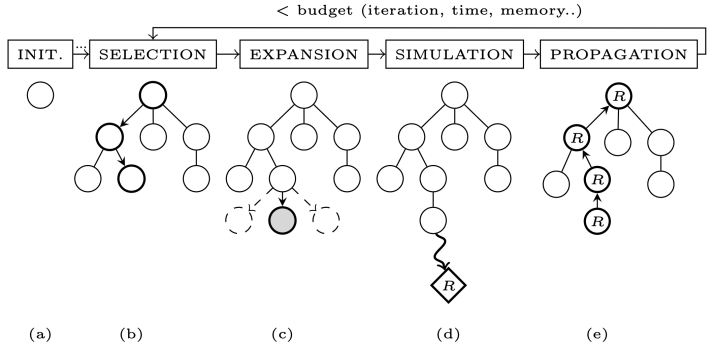


Fig. 13. Principes de la recherche arborescente Monte Carlo (MCTS).

Une des difficultés majeures pour utiliser MCTS est la stratégie à adopter dans la phase de sélection (figure 13.b) où il faut choisir entre un nœud qui est prometteur (exploitation) et un nœud pour lequel peu de simulations ont été réalisées (exploration). Par défaut, MCTS utilise les limites supérieures de confiance appliquées aux arbres (*upper confidence bounds for trees*, UCT) mais l'heuristique *Rapid action value estimation* (RAVE) s'est rapidement imposée à la fois dans le domaine discret et le domaine continu (cRAVE) ; elle assure un partage de connaissances entre des nœuds apparentés, ce qui permet d'obtenir une estimation rapide mais biaisée des valeurs d'action. Ensuite, une généralisation de RAVE (GRAVE) a été proposée pour recueillir des estimations plus précises près des feuilles de l'arbre mais uniquement dans le cas discret [7]. Ainsi, notre première contribution dans [22] a été d'adapter GRAVE au domaine continu (cGRAVE). Ensuite, nous avons suggéré de décomposer une action multidimensionnelle, une célérité qui est définie sur plusieurs dimensions, en plusieurs actions unidimensionnelles (AD), ce qui a pour conséquence d'augmenter la hauteur de l'arbre. La dernière contribution consiste à utiliser les contraintes pour réduire *a priori* l'espace des possibilités pour les actions (*constraint-based selective policy*, CSP). Nous avons comparé les performances de ces différentes approches (figure 14) en testant à chaque fois l'apport de chaque contribution (CSP, AD ou cGRAVE) à la technique de l'état de l'art (cRAVE).

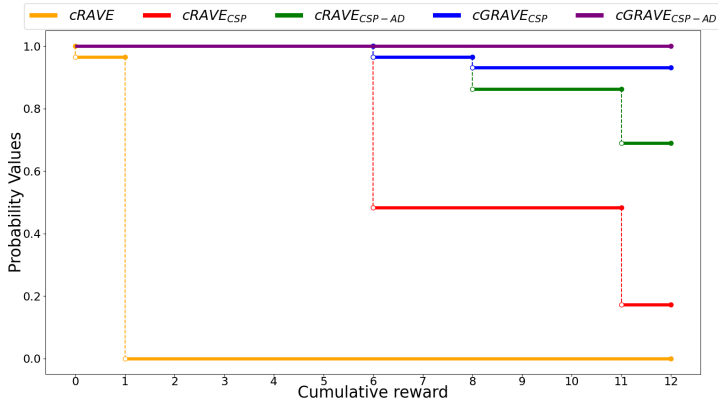


Fig. 14. Fonctions de répartition de plusieurs stratégies d'apprentissage par renforcement continu sur la modélisation d'un RRBh à 5 gènes sur 30 exécutions. Plus le gain est grand, avec un maximum à 12, meilleure est la solution.

On remarque que cRAVE combinée ou non à une stratégie de gestion de contraintes (cRAVE$_{CSP}$) n'arrive pas à obtenir de solutions comme pour les problèmes de satisfaction de contraintes continues même si la stratégie CSP améliore grandement le gain obtenu, passant de 1 à 6 dans la quasi-totalité des cas. Par rapport à cRAVE$_{CSP}$, la stratégie de décomposition AD (cRAVE$_{CSP-AD}$) de faire mieux que tripler la probabilité d'obtenir des solutions viables, passant de 0.2 à presque 0.7. De plus, notre adaptation de GRAVE au domaine continu (cGRAVE$_{CSP}$) atteint une probabilité proche de 1 d'obtenir une solution viable comparée à cRAVE$_{CSP}$. C'est seulement la combinaison des trois propositions (cGRAVE$_{CSP-AD}$) qui permet d'obtenir des échantillons respectant toutes les contraintes à coup sûr (probabilité égale à 1).

Multiples solutions ?

Comme pour les métaheuristiques à base de population, la grande majorité des variantes de l'apprentissage par renforcement basées sur MCTS ne permettent d'obtenir qu'une seule séquence optimale d'actions à réaliser. Afin de proposer un ensemble diversifié de solutions, nous avons proposé trois contributions différentes permettant d'obtenir plusieurs séquences optimales dans le domaine continu [21] en se basant sur les récents travaux de [4] qui formalisent parfaitement la problématique dans le domaine discret (*diverse multi-armed bandit policies*, DMAB). Nous introduisons deux stratégies d'inhibition différentes (*lock* et *diverse progressive widening*, DivPW) renforçant la diversité pendant la construction de l'arbre de recherche ainsi qu'une troisième (*diverse planning with MO-MCTS*, DPMO) qui ajoute un

deuxième objectif de diversité à une variante multiobjectif de MCTS [28]. Nous avons testé ces stratégies sur notre problème d'identification de célérités. Nous pouvons affirmer que sur un RRBh à 3 gènes (figure 15, à gauche) DivPW surpasse largement les autres stratégies testées. DPMO et Lock obtiennent de meilleures performances en matière de diversité que DMAB, mais leurs performances ne sont pas statistiquement différentes. Sur un RRBh à 5 gènes (figure 15, à droite), la stratégie de référence DMAB est également à la traîne par rapport aux stratégies proposées. Cependant, DivPW est surpassée par DPMO. Dans l'ensemble, cela démontre l'intérêt des nouvelles stratégies proposées.

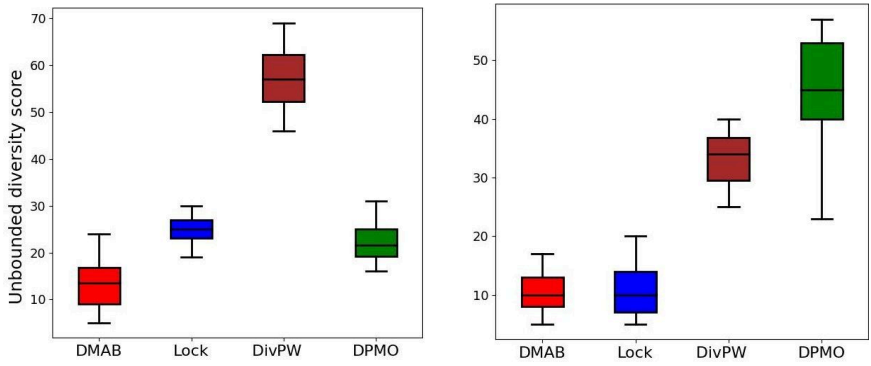


Fig. 15. Boîtes à moustaches du score de diversité moyen de nos stratégies par rapport à DMAB sur un RRBh à 3 (à gauche) et 5 gènes (à droite).

Conclusion et perspectives

Dans ces travaux, nous démontrons l'intérêt de l'évolution artificielle, par le biais des métaheuristiques à base de population, et de l'apprentissage par renforcement, via une recherche arborescente Monte-Carlo, pour résoudre la paramétrisation de réseaux de régulation biologique (RRB) par rapport aux solveurs de contraintes continues (cCSP). Contrairement à de nombreux travaux de recherche qui infèrent des RRB, nous adaptons les deux approches pour construire des RRB discrets et hybrides plus finement. De plus, nous proposons, dans les deux cas, des stratégies qui autorisent plusieurs paramétrisations optimales, intéressantes pour les biologistes. Dans le cas de l'évolution artificielle, nous mettons même en exergue certaines lacunes des fonctions de tests de la compétition CEC'2020 [17] sur l'optimisation multimodale qui considère l'identification d'un seul optimum

par bassin d'attraction alors que notre cadre d'étude exhibe un cas où il y en a une infinité. Dans le cas de l'apprentissage par renforcement, nous proposons même une adaptation d'une récente version de MCTS (GRAVE) au domaine du continu ainsi que plusieurs contributions pour obtenir plusieurs séquences optimales diversifiées (Lock, DPMO, DivPW).

On pourrait être tenté de comparer les métaheuristiques à MCTS sur les mêmes problèmes (RRBh à 2, 3 puis 5 gènes) mais cela ne nous semble pas équitable dans la mesure où une trajectoire complète est évaluée pour l'évolution artificielle alors qu'elle est construite de manière incrémentale pour MCTS. De plus, dans le cas de MCTS, les contraintes biologiques sont utilisées pour réduire l'ensemble des valeurs de la célérité dans chaque espace discret. Une démarche similaire pour les métaheuristiques serait de construire des opérateurs génétiques (croisement, mutation) capables de produire uniquement des trajectoires viables qui respecteraient les contraintes biologiques. Cela pourrait être une nouvelle piste de recherche. Une autre piste de recherche intéressante est de valider nos différentes contributions de MCTS (pour le domaine continu et pour les séquences diversifiées) sur des problèmes plus classiques de la communauté. Enfin, une dernière piste que nous envisageons consiste à adapter les techniques d'optimisation mixte [27] qui gèrent de manière plus appropriée le fait que nous manipulons simultanément des entiers et des réels.

Références

[1] A. Ahrari, S. Elsayed, R. Sarker D. Essam, et C. C. Coello. 2022. Static and Dynamic Multimodal Optimization by Improved Covariance Matrix Self-Adaptation Evolution Strategy With Repelling Subpopulations. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2021.3117116.

[2] E. Alba et B. Dorronsoro. 2008. Introduction to cellular genetic algorithms. In *Cellular Genetic Algorithms*. https://doi.org/10.1007/978-0-387-77610-1_1.

[3] J. Behaegel, J.-P. Comet, et M. Pelleau. 2018. Identification of Dynamic Parameters for Gene Networks. In *ICTAI*. https://doi.org/10.1109/ICTAI.2018.00028.

[4] L. Benke, T. Miller, M. Papasimeon, et N. Lipovetzky. 2023. Diverse, Top-k, and Top-Quality Planning Over Simulators. https://doi.org/10.3233/FAIA230275.

[5] G. Bernot, J.-P. Comet, Z. Khalis, A. Richard, et O. Roux. 2019. A genetically modified Hoare logic. *Theoretical Computer Science*. https://doi.org/10.1016/j.tcs.2018.02.003.

[6] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, et S. Colton. 2012. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in Games*. https://doi.org/10.1109/TCIAIG.2012.2186810.

[7] Tristan Cazenave. 2015. Generalized rapid action value estimation. In (IJCAI). Consulté à l'adresse https://dl.acm.org/doi/abs/10.5555/2832249.2832354.

[8] J. Dong, J. Li, et F. Wang. 2024. Deep Learning in Gene Regulatory Network Inference: A Survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. https://doi.org/10.1109/TCBB.2024.3442536.

[9] J. M. Escorcia-Rodríguez, E. Gaytan-Nuñez, E. M. Hernandez-Benitez, A. Zorro-Aranda, M. A. Tello-Palencia, et J. A. Freyre-González. 2023. Improving gene regulatory network inference and assessment: The importance of using network structure. *Frontiers in Genetics*. https://doi.org/10.3389/fgene.2023.1143382.

[10] G. Grataloup, J.-P. Comet, et G. Bernot. 2024. Improving Snoussi constraints in the Thomas framework for Gene Networks. In *Journées ouvertes de biologie, informatique et mathématiques*.

[11] G. Grataloup, D. Pallez, J.-P. Comet, et G. Bernot. 2024. Single-objective constrained optimization for Gene Regulatory Networks Modeling. In *EA*.

[12] J. Gurrola-Ramos, A. Hernàndez-Aguirre, et O. Dalmau-Cedeño. 2020. COLSHADE for Real-World Single-Objective Constrained optimization Problems. In (CEC). https://doi.org/10.1109/CEC48606.2020.9185583.

[13] V. A. Huynh-Thu et G. Sanguinetti. 2019. Gene Regulatory Network Inference: An Introductory Survey. In *Gene Regulatory Networks: Methods and Protocols*. https://doi.org/10.1007/978-1-4939-8882-2_1.

[14] M. Kronfeld et A. Zell. 2010. Towards scalability in niching methods. In (CEC). https://doi.org/10.1109/CEC.2010.5585916.

[15] A. Kumar, S. Das, et I. Zelinka. 2020. A self-adaptive spherical search algorithm for real-world constrained optimization problems. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion* (GECCO). https://doi.org/10.1145/3377929.3398186.

[16] A. Kumar, S. Das, et I. Zelinka. 2020. A modified covariance matrix adaptation evolution strategy for real-world constrained optimization problems. In (GECCO). https://doi.org/10.1145/3377929.3398185.

[17] A. Kumar, Guohua Wu, Mostafa Z. Ali, Rammohan Mallipeddi, Ponnuthurai Nagaratnam Suganthan, et Swagatam Das. 2020. A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2020.100693.

[18] X. Li, A. Engelbrecht, et M. G. Epitropakis. 2013. Benchmark Functions for CEC2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization. Consulté à l'adresse https://titan.csit.rmit.edu.au/~e46507/cec13-niching/competition.

[19] R. Michelucci, V. Callegari, J.-P. Comet, et D. Pallez. 2024. Cellular Genetic Algorithms for identifying variables in hybrid Gene Regulatory Networks. In *EvoAPPS*. https://doi.org/10.1007/978-3-031-56852-7_9.

[20] R. Michelucci, J.-P. Comet, et D. Pallez. 2022. Evolutionary continuous optimization of hybrid Gene Regulatory Networks. In *EA*. https://doi.org/10.1007/978-3-031-42616-2_12.

[21] R. Michelucci, J.-P. Comet, et D. Pallez. 2024. Comparing diverse planning strategies with continuous MCTS applied to hGRN. In *ICTAI*. https://doi.org/10.1109/ICTAI62512.2024.00018

[22] R. Michelucci, D. Pallez, T. Cazenave, et J.-P. Comet. 2024. Improving Continuous Monte Carlo Tree Search for Identifying Parameters in Hybrid GRN. In *PPSN*. https://doi.org/10.1007/978-3-031-70085-9_20.

[23] R. Michelucci, D. Pallez, et J.-P. Comet. 2025. Parametrisation of hybrid Gene Regulatory Networks using Artificial Evolution and Reinforcement Learning. In *ROADEF*. Consulté à l'adresse https://inria.hal.science/hal-04980785.

[24] H. Nguyen, D. Tran, B. Tran, B. Pehlivan, et T. Nguyen. 2020. A comprehensive survey of regulatory network inference methods using single cell RNA sequencing data. *Briefings in Bioinformatics*. https://doi.org/10.1093/bib/bbaa190.

[25] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Driessche, T. Graepel, et D. Hassabis. 2017. Mastering the Game of Go without Human Knowledge. *Nature*. https://doi.org/10.1038/nature24270.

[26] Maciej Świechowski, Konrad Godlewski, Bartosz Sawicki, et Jacek Mańdziuk. 2023. Monte Carlo Tree Search: A Review of Recent Modifications and Applications. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-022-10228-y.

[27] El-Ghazali Talbi. 2024. Metaheuristics for variable-size mixed optimization problems: A unified taxonomy and survey. *Swarm and Evolutionary Computation*. https://doi.org/10.1016/j.swevo.2024.101642.

[28] Weijia Wang et Michèle Sebag. 2012. Multi-objective Monte-Carlo Tree Search. In *Proceedings of the Asian Conference on Machine Learning*. Consulté à l'adresse https://inria.hal.science/hal-00758379.

L'ordinateur et ses périphériques

Une course d'obstacles épistémologiques

Emmanuel Saint-James
Sorbonne Université

Par trois fois, l'histoire de l'informatique a été présentée comme une succession de générations : de processeurs, de machines, de langages de programmation. Ces considérations technologiques, non dénuées d'intérêts économiques plus ou moins avoués, sont irrecevables et la première partie de cet article montrera pourquoi. La deuxième partie exposera une vision fondée sur la notion bachelardienne d'obstacle épistémologique, en percevant dix obstacles qui furent franchis en adjoignant à l'*unité arithmétique et logique* de nouveaux *périphériques*, débouchant non seulement sur des alliages, au sens deleuzien, inédits, mais aussi sur des branches théoriques nouvelles de la discipline. Ainsi, par un étonnant paradoxe, ce qui est central dans l'évolution des ordinateurs est ce qui en est à la périphérie. Toutefois, une partie de la discipline n'apparaît pas dans cette perspective, ce qui s'explique par un statut épistémologique en fait différent, cette branche entretenant avec la science informatique un rapport semblable à celui de la médecine avec la biologie[1].

1. Les idées du présent article ont d'abord été exposées à une conférence de 2022 dans le cadre des 75 ans d'informatique en France. Filant la métaphore entre cette course de 900 mois semée de périphériques et le 400 mètres-haies, comportant lui aussi dix obstacles, cette conférence et son support iconographique animé, ironiquement sportif, est disponible à cette adresse : `https://www.lip6.fr/75ans/?guest=Saint-James`.

Je remercie Jean-Luc Mounier pour ses encouragements à élaborer cette conférence, et je remercie Éric Guichard pour son invitation à la réexposer à l'ENS de Lyon.

DOI:10.48556/SIF.1024.26.101

Les générations perdues

Les générations de langages

En première approche, l'histoire des langages de programmation est essentiellement une suite de *dispenses de codage* après le premier d'entre eux :

1. les langages binaires des processeurs imposent le codage des instructions, des adresses en mémoire, des structures de données et de l'algorithme ;
2. les langages d'assemblage, version alphabétique des précédents, dispensent du codage des instructions ;
3. les langages de programmation dits impératifs, type FORTRAN, dispensent du codage des expressions arithmétiques ;
4. les langages de programmation dits fonctionnels (ou applicatifs selon la terminologie de l'époque), type LISP, dispensent du codage des données ;
5. les langages de programmation dits logiques (ou déclaratifs), type PROLOG, dispensent du codage de l'algorithme.

Cette suite appelle quelques remarques :
– le codage manuel en binaire ne subsiste pratiquement plus que chez les constructeurs de processeurs ;
– l'écriture directe en langage d'assemblage est devenue marginale ;
– chaque génération permet d'exprimer plus brièvement que la précédente une même intention.

Fort de ces évolutions, le cinquième terme de la suite a été vu comme devant fatalement marginaliser voire éliminer ses prédécesseurs. Pourtant, c'est lui qui ne leur a pas survécu. Ce pronostic s'est révélé erroné parce qu'il ne tenait pas compte d'autres phénomènes :
– les influences mutuelles :
  → la *récursivité* et les *fonctions anonymes*, naguère considérées comme des bizarreries déraisonnables des langages fonctionnels, sont disponibles dans nombre de langages impératifs d'aujourd'hui, adoptant même le *passage à la continuation* pourtant né dans un langage fonctionnel dynamiquement typé,
  → la reprise sur échec, principe central de la programmation logique, est disponible dans beaucoup de langages, via une interface avec PCRE[2], l'interprète d'expressions rationnelles décrivant des *motifs*,
  → les mécanismes d'*instanciation et d'héritage de classe*, ont été adoptés par toutes les familles de langages (montrant au passage qu'il s'agit d'une méthodologie universelle et non d'une nouvelle famille de langages comme cela est souvent présenté) ;
– le besoin de langages à l'expressivité volontairement restreinte :

2. *Perl Compatible Regular Expressions*.

→ renoncement à la complétude algorithmique, notamment dans les langages d'interrogation de bases de données comme SQL,
→ rejet d'expressions non typables statiquement, en ADA ou ML et ses successeurs,
→ restriction des entrées-sorties dans les langages embarqués comme Javascript,
→ la réflexivité, facilitée en Lisp par l'identité entre programmes et données, est découragée, voire interdite.

En définitive, les langages de programmation tendent vers une certaine homogénéité, mais en restreignant l'expressivité. Celle-ci n'est plus la priorité, la sécurité l'a remplacée.

Les générations de machines

Les ordinateurs reposent sur des architectures différentes, pourtant il a été proposé un découpage générationnel trompeur, qui a d'ailleurs fini par tourner court :

1. première génération, utilisation de tubes à vide ;
2. deuxième génération, utilisation de transistors ;
3. troisième génération, utilisation de circuits intégrés (câblage de plusieurs transistors) ;
4. quatrième génération, utilisation de circuits très intégrés (changement qui n'est plus que quantitatif) ;
5. un projet dit de *cinquième génération* lancé en 1982 promettait en dix ans de résoudre par une *machine Prolog* le problème de la lenteur de ce langage ; mais ce projet :
   – n'a abouti qu'à un nouveau compilateur Prolog, aucune machine n'ayant jamais été décrite et encore moins réalisée ;
   – n'avait pas médité l'échec des *machines Lisp*, qui avaient déjà reporté sur le matériel le problème, également présent en Prolog, de la récupération de mémoire, d'où un gain indéniable mais insuffisant pour rendre compétitives ces machines spécialisées[3].

Cette cinquième génération annoncée par le Japon a tellement inquiété les pouvoirs publics européens qu'il se précipitèrent pour financer un projet apparamment concurrent. Le fiasco qui en résulta, typique de ce qui se passe

3. L'ICOT, responsable de ce projet, éditait un journal dont seuls les sommaires ont été mis en ligne ; le numéro 24 de 1989 (`https://www.airc.aist.go.jp/aitec-icot/ICOT/Museum/JOURNAL/icot-jrnl24.html`), possédé par l'auteur, suggère déjà qu'il s'agissait d'un mauvais choix, au regard du *neuro boom* (sic) apparu depuis 1982.

lorsque le pouvoir politique s'immisce dans la dynamique de la recherche scientifique s'appelait *machines à réduction*[4] :

– l'idée était d'abandonner l'architecture dite « de von Neumann », au profit des modèles mathématiques du calcul antérieurs à l'ordinateur ;
– ces modèles, le *lambda-calcul* et son dérivé la *logique combinatoire*, reposent sur un mécanisme de *réduction*, consistant à remplacer l'application d'une fonction sur un argument par une version spécialisée de cette fonction pour cet argument ;
– cette réécriture induit des copies incessantes à un coût faramineux, raison pour laquelle les pionniers de l'informatique se sont empressés d'inventer :
  → l'*adresse en mémoire*, réalisation concrète de la notion abstraite de variable en mathématique,
  → le *registre d'adresse*, permettant de substituer une variable à sa valeur par une indirection via un registre d'adresse, non plus par réécriture de tout le programme ;

Ce projet de machines à réduction n'aura abouti qu'à une réduction de ses crédits. En l'occurrence, la montagne n'aura même pas accouché d'une souris, puisque, comme on le verra, c'est justement l'invention de la souris d'ordinateur qui sera la vraie révolution de cette époque.

Les générations de processeurs

Connue sous le nom de « loi de Moore », pionnier de la fabrication de microprocesseurs, une autre perspective concerne l'évolution de ceux-ci, faussée cette fois par des problèmes de vocabulaire :

– en science, le mot « loi » s'applique à des phénomènes naturels, l'utiliser pour un objet manufacturé est usurper la confiance mise dans ce mot dans les sciences de la nature (un choix peu surprenant dans une langue désignant l'informatique par « *computer science* » qui institue cette usurpation) ;
– elle a connu plusieurs énoncés[5] :
  → en 1965, que le nombre de transistors sur un circuit intégré doublait tous les ans,
  → en 1975, que ce ne serait finalement que tous les deux ans,
  → et finalement que ce serait la « performance des ordinateurs » qui doublerait tous les 18 mois, glissements de sens révélateur :

4. Une description peut être trouvée dans Simon Peyton Jones : *GRIP—a high-performance architecture for parallel graph reduction* in Conference on Functional Programming Languages and Computer Architecture LNCS274 1987 Springer, également disponible à l'adresse `https://www.microsoft.com/en-us/research/wp-content/uploads/1987/09/GRIP.pdf`, où on peut constater que la récupération de la mémoire repose sur la solution classique du marquage bloquant.

5. Voir Christophe Lécuyer et Hyungsub Choi *Les secrets de la Silicon Valley ou les entreprises américaines de microélectronique face à l'incertitude technique* Revue d'histoire moderne et contemporaine 2012/3 n° 59-3 Belin.

→ il ne s'agit plus du seul processeur mais de l'ordinateur,
→ il ne s'agit plus de taux d'intégration mais de « performance », mot suffisamment vague pour permettre de postuler ce qu'on veut.

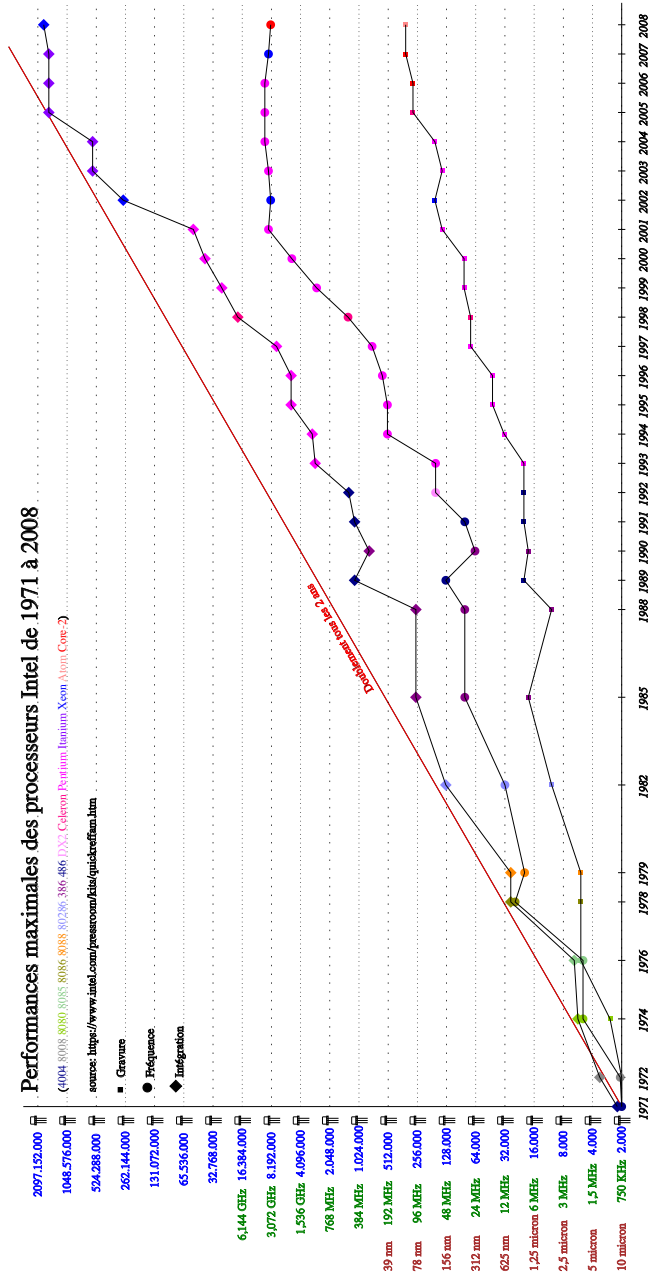
Le site Web officiel de l'entreprise concernée fournit des tableaux de chiffres sur ses processeurs de 1971 à 2008[6]. Le graphique de la page suivante les présente sur une échelle logarithmique afin que la diagonale du repère représente le doublement tous les 2 ans. Il montre que le nombre de transistors intégrés dans un processeur suit, quoique légèrement en dessous, cette diagonale, mais que depuis 2001 l'odyssée de la fréquence et de la gravure stagne.

Cette prétendue loi est donc moins trompeuse que les deux perspectives précédentes, mais elle obscurcit voire ignore les évolutions qui ont crédibilisé ces discours sur le doublement des performances :

– la plus grande finesse de gravure a permis en fait :
  → l'accélération des calculs parce que le chemin parcouru par les électrons est plus court,
  → l'adjonction au processeur de mémoires dites *cache*, en plus des registres,
  → la multiplication des unités arithmétiques et logiques dans le processeur (les *multicœurs*) ;
– l'adoption d'une longueur identique pour toutes les instructions d'où :
  → le chargement de toute instruction, opérandes compris, en un seul cycle d'horloge,
  → la parallélisation des étapes (de 2 à 31 selon les processeurs) de traitement des instructions par un pipeline ayant autant d'étages que d'étapes, ce qui permet d'approcher l'égalité entre fréquence d'horloge et nombre d'instructions exécutées par seconde (quand un pipeline de n étages est plein, n étapes sont exécutées pour n instructions en 1 cycle d'horloge, soit l'équivalent d'une seule instruction exécutée en 1 cycle) ;
  → le branchement prédictif, qui augmente les moments où le pipeline est plein.

Les informations sur les années postérieures à 2008 ne sont plus fournies sous forme tabulaire, trop de choses ayant changé depuis. La fréquence d'horloge, en particulier, n'est plus une constante : elle varie automatiquement en fonction de l'activité des cœurs, afin d'éviter des surchauffes ainsi qu'une consommation excessive lors d'un fonctionnement sur batterie. Le nombre de transistors a cessé de croître exponentiellement : seulement une dizaine de fois plus par décennie. La gravure a continué à s'affiner, s'approchant du nanomètre, mais ce progrès est moins dû à l'industrie de l'informatique, à strictement parler, qu'à celle de la photolithographie, notamment l'entreprise hollandaise ASML dont les machines produisent les *wafers* constitutifs de la plupart des processeurs modernes.

6. https://www.intel.com/pressroom/kits/quickreffam.htm.


Performances maximales des processeurs Intel de 1971 à 2008
(4004 8008 8080 8085 8086 8088 80286 386 486 DX2 Celeron Pentium Itanium Xeon Atom Core 2)
source : https://www.intel.com/pressroom/kits/quickreffam.htm
Gravure
Fréquence
Intégration
Doublement tous les 2 ans
2097.152.000
1048.576.000
524.288.000
262.144.000
131.072.000
65.536.000
32.768.000
16.384.000
8.192.000
4.096.000
2.048.000
1.024.000
512.000
256.000
128.000
64.000
32.000
16.000
8.000
4.000
2.000
6,144 GHz
3,072 GHz
1,536 GHz
768 MHz
384 MHz
192 MHz
96 MHz
48 MHz
24 MHz
12 MHz
6 MHz
3 MHz
1,5 MHz
750 KHz
39 nm
78 nm
156 nm
312 nm
625 nm
1,25 micron
2,5 micron
5 micron
10 micron
1971 1972 1974 1976 1978 1979 1982 1985 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008

Sans nier l'importance des investissements financiers sur le plan matériel, il faut noter que l'amélioration générale est aussi due à l'investissement intellectuel ayant abouti à des idées nouvelles (pipeline et multicœurs). L'opposition maladroite des dénominations RISC (reduced instruction set computer) et CISC (complex instruction set computer) a masqué que la question centrale est la longueur unique des instructions, car elle permet le pipeline, et non la longueur des jeux d'instructions (ceux des premiers processeurs étaient d'ailleurs encore plus réduits que ceux des victorieux processeurs RISC).

Obstacle épistémologique et alliage innovant

Les trois perspectives citées avaient pour but de justifier des investissements dans une direction bien précise. Ce point de vue inévitablement biaisé n'est pas utilisé en histoire des sciences, qui s'est forgé des concepts plus profonds :

– la notion d'*obstacle épistémologique*, élaborée par Gaston Bachelard[7], qui empêche d'étudier scientifiquement un phénomène, notamment les préjugés inconscients qui se sont glissés dans la connaissance générale ;
– surmonter ces obstacles permet d'effectuer une *rupture épistémologique* dans la compréhension d'un phénomène ;
– des auteurs plus récents ont parlé de *changement de paradigme*, notion équivalente ; à l'inverse, on peut voir chez Léon Brunschvicg[8] une description avant la lettre de ruptures épistémologiques dans les élargissements successifs de la notion de nombre en mathématique (entiers, fractions, irrationnels, complexes) ;
– enfin, la notion d'*alliage*, bien connue en métallurgie mais que le philosophe Gilles Deleuze a étendue[9].

L'histoire de l'informatique se révèle ponctuée d'alliages nouveaux entre le processeur et un *périphérique*. La thèse ici défendue est que l'histoire de l'informatique est essentiellement une suite de franchissements d'obstacles épistémologiques permettant des alliages entre le processeur et un périphérique nouveau ou issu d'un autre univers technologique. Chacun de ces franchissements suscite l'élaboration ou le rajeunissement de méthodologies voire de théories qui finissent par déborder leur inspiration matérielle originelle.

7. Gaston Bachelard. *La formation de l'esprit scientifique*, Paris, Vrin, 1938.
8. Léon Brunschvicg. *Les étapes de la philosophie mathématique*, Paris, Félix Alcan, 1912.
9. Gilles Deleuze et Félix Guattari. *Mille Plateaux*, Paris, Les Éditions de minuit, 1980.

La mémoire

Il semble aujourd'hui évident que la mémoire soit indispensable à un ordinateur. Cependant :
– un des pionniers du calcul automatisé considérait que l'utilisation d'une mémoire conduirait nécessairement à un ralentissement des calculs et a cherché une autre voie même après la diffusion des premiers ordinateurs[10] ;
– ici l'obstacle était de s'aligner sur l'épistémologie de la physique et non des mathématiques :
  → pour la physique, *connaître c'est mesurer* (avec des instruments de mesure),
  → pour les mathématiques, *connaître c'est abstraire* (avec des variables).

L'irruption de la mémoire artificielle a sorti de l'enfance la balbutiante *algorithmique*. Réduire le nombre d'opérations dans une méthode de calcul est une vieille idée, mais pas l'optimisation de la gestion des résultats intermédiaires à garder visibles et rapidement accessibles sur le support d'écriture. L'algoritmique de l'*analyse numérique*, en particulier, s'est complètement renouvelée avec la question des erreurs d'arrondis en base 2, et celle du recours aux représentations en virgule fixe ou en virgule flottante selon les situations.

Le clavier

Avant que l'on songe à emprunter son clavier à la machine écrire (pourtant apparue dès le XVIII^e siècle), les périphériques d'entrée étaient :
– soit des potentiomètres, pour les calculateurs analogiques ;
– soit des câbles à enfoncer dans des prises pour relier deux circuits électriques ;
– soit des boutons-poussoirs reliant ou déconnectant deux circuits.

Dans tous les cas, l'idée était que calculer était une opération manuelle de *configuration*. *Programmer* (verbe inventé à cette époque), c'est obtenir un acte à partir d'un texte, ce qui récuse la dichotomie traditionnelle entre la pensée et l'action. Cette irruption du texte en informatique est largement à l'origine de la *mathématisation de la linguistique*, avec les *grammaires génératives* et la renaissance des *théories des types de données*. Les premières relèvent de la syntaxe, mais une syntaxe strictement linéaire interdisant indices, exposants et plus généralement tout effet de mise en pages. Les deuxièmes relèvent non pas, comme l'affirme sans retenue le vocabulaire informatique, de la sémantique, mais de la *signification*, les ordinateurs ne faisant que manipuler des systèmes de signes dont le *sens* leur échappe. Les

10. Girolamo Ramunni. *Louis Couffignal (1902-1966) : un pionnier de l'informatique en France*, Colloque sur l'histoire de l'informatique en France, Grenoble, 1988, AFCET/CNRS.

théories des types garantissent seulement une *cohérence* dans les opérations sur les données, au prix d'une perte d'expressivité comme cela a été signalé.

Le stockage

Initialement, le clavier servait à perforer des cartes, support utilisé notamment pour collecter des données lors des recensements et permettre leur exploitation statistique. Ce précédent longtemps vivace dans l'industrie informatique a freiné le développement des mémoires de stockage électrifiées : le recensement est une opération d'*archivage*, qui est une forme restreinte de *mémorisation* (mot récent, employé d'abord en didactique) en ce qu'elle ne permet pas de modifier ce qui a été mémorisé, usage sans intérêt et même nuisible à l'archivage.

La disponibilité de supports de stockage réinscriptibles et fiables a donné naissance aux versions informatisées des *fichiers*, des *répertoires* et des *bases de données*. Ces dernières ont suscité l'invention de langages spécifiques d'interrogation et d'actualisation, reposant sur des algorithmes sophistiqués.

Le contrôleur d'interruptions

Le contrôleur d'interruptions a permis de remplacer les demandes incessantes et souvent vaines du processeur aux périphériques, par des envois toujours utiles de ceux-ci vers celui-là. Cette nouvelle architecture n'a été possible qu'en renonçant à l'idée qu'une pensée bien construite est nécessairement linéaire. La lecture séquentielle d'un programme utilisant des interruptions n'est alors plus le reflet de la séquence d'instructions que provoquera son exécution : à tout moment une interruption peut lancer l'exécution d'un *preneur d'événement* à la place du processus en cours d'exécution, provisoirement abandonné puis réexécuté ultérieurement à partir de là où il avait été suspendu.

Cette possibilité d'interruption, en particulier par l'horloge pour forcer le processeur à cesser d'exécuter un processus pour se consacrer à un autre lui-même précédemment suspendu, est à l'origine du développement théorique et méthodologique des *systèmes d'exploitation des ordinateurs*, notamment leur *ordonnanceur* permettant le *temps partagé*.

L'écran alphanumérique

Le remplacement des bruyantes machines à écrire par de silencieux écrans était une évidence dont la réalisation a été ralentie par des raisons budgétaires, mais pas seulement. Le gain ergonomique potentiel n'a pas été immédiatement perçu, rédiger un texte étant une activité apparemment pas susceptible des mêmes interrogations sur la fatigue au travail qu'un métier manuel.

C'est à partir de ce moment que s'est développé la branche de l'informatique nommée « interface homme machine », incluant l'ergonomie du logiciel et plus tard le *Web-design*, une branche très liée à l'évolution des périphériques.

Le réseau

L'interconnexion des ordinateurs a été freinée d'abord par la volonté des constructeurs d'ordinateurs mais aussi d'appareils de télécommunications, de garder captives leurs clientèles respectives. Toutefois le plan épistémologique présentait un double obstacle :

– la tendance à assimiler sécurité et inaccessibilité physique, qui a été dépassée :
  → par un renouveau de la *cryptographie*, bénéficiant des avancées de l'algorithmique,
  → par une mathématisation poussée des protocoles de (télé)communication, notamment avec les *codes correcteurs d'erreurs* permettant la fiabilité d'*Internet* ;
– la difficulté à constater qu'un système d'exploitation passe moins de temps à gérer des *fichiers* que des *flux de données* ; cette vision dynamique et non plus statique :
  → a débouché sur l'invention des canaux de communication (le *pipe* et la *socket*) par le système d'exploitation Unix (conçu par une entreprise de téléphonie, non pas d'informatique, ce n'est pas un hasard), permettant la réalisation des algorithmes perpétuels que sont les *serveurs*,
  → a renouvelé la recherche dans les systèmes d'exploitation, les concurrents d'Unix étant obligés de le copier, avec POSIX, ou de disparaître.

L'imprimante

L'imprimante a été un périphérique déterminant dans l'histoire de l'informatique. Avant son apparition, un dirigeant d'un grand constructeur informatique aujourd'hui disparu déclarait : « *je ne vois aucune bonne raison pour un particulier d'avoir un ordinateur chez lui* ». Réflexion surprenante maintenant que chacun a dans sa poche un ordinateur servant aussi de téléphone, mais pas dénuée de fondement à l'époque. En effet, le premier micro-ordinateur, comme nombre d'autres inventions françaises, a été un échec faute de comprendre où était le marché. Celui-ci n'est devenu évident qu'avec son *alliage* avec une imprimante individuelle, qui a permis de capter le marché de la machine à écrire. La confusion ici était entre la *découverte scientifique* ou l'*invention technologique*, qui procèdent d'une nouveauté intrinsèque au domaine concerné, et l'*innovation*, laquelle est définie par les économistes comme *ce qui provoque un déplacement de la demande sur un marché*, ici de la machine à écrire au départ, ensuite de tout le secteur de l'imprimerie avec les imprimantes laser.

De là sont nées les recherches en *typographie numérique*, qui seront reprises par les cartes graphiques ultérieures. Ce nouveau modèle graphique est appelé *vectoriel* car il remplaçait la grille de points par des vecteurs, aujourd'hui eux-mêmes remplacés par courbes de Bézier ; mais le terme est resté.

Les périphériques gestuels

Une multitude de nouveaux périphériques apparaît ensuite, dont la caractéristique commune est d'aller par paire ou plus : l'un capte, l'autre restitue, des mouvements d'un objet ou d'un être vivant, et même sa voix et son souffle. Naissent simultanément la souris et l'écran à pixels séparés (dit *écran graphique* par opposition à *écran alphanumérique*), les haut-parleurs intégrés à l'ordinateur couplés à un micro, une caméra ou un instrument de musique MIDI. Les documents gérés par ces périphériques auraient pu être regroupés sous la dénomination de *documents gestuels*, ce qui n'est pas le cas dans l'incons(is)tante, non versionnée et indéboulonnable nomenclature MIME (multipurpose Internet mail extensions) malgré sa définition bien postérieure à eux. Cette incohérence témoigne de l'obstacle épistémologique rencontré :

– penser est plus général que verbaliser ; ce qui se conçoit bien ne s'énonce pas nécessairement par des mots ; les compositeurs, les architectes, les dessinateurs et les sculpteurs le savent bien ;
– les branches de l'informatique induites par ces périphériques sont l'analyse et la synthèse des sons, des images et des mouvements (informatique musicale, reconnaissance des formes, traitement du signal, robotique mais aussi algorithmes de compression) et cette liste s'allonge avec les nouveaux capteurs qui ne cessent d'apparaître, en informatique médicale notamment.

Les coprocesseurs

La présence de plusieurs processeurs dans un ordinateur offre la possibilité de calculs parallèles, terme qui a été employé dans des situations en fait différentes :

– les systèmes d'exploitation à temps partagé sont confrontés à un apparent parallélisme, vécu d'abord comme une gêne, résolue par des mécanismes de verrouillage d'accès, notamment le *sémaphore* pour éviter les *étreintes fatales* ;
– de l'adjonction de coprocesseurs numériques ou graphiques a germé l'idée qu'un processeur pouvait en commander un autre ;
– admettre que des processeurs pouvaient être vus comme périphériques d'un autre aussi puissant (voire moins puissant) était l'obstacle à franchir pour déboucher sur l'*algorithmique répartie*, sans laquelle les *moteurs de recherche du Web* ne pourraient faire face à l'évolution constante des sites Web ;

Cette difficulté à accepter les calculs parallèles pouvait être surmontée en prenant des leçons de musique, les différentes architectures d'ordinateurs n'étant pas sans évoquer certaines formes musicales :
– *Single Instruction Single Data* : monophonie ;
– *Multiple Instruction Single Data* : polyphonie classique ;
– *Single Instruction Multiple Data* : polyphonie romantique ;
– *Multiple Instruction Multiple Data* : polytonalité, polymodalité, atonalité.

Le périphérique distant

La rupture se situe ici dans l'idée qu'un périphérique n'est pas nécessairement relié physiquement et exclusivement à un processeur : les satellites d'un système de positionnement émettent un signal disponible pour qui veut, et doivent être vus comme des périphériques partagés par tous les ordinateurs capables d'interpréter ce signal.

La mobilité, déjà esquissée avec le protocole HTTP qui fonctionne en mode déconnecté contrairement à ses prédécesseurs (FTP, SMTP, IMAP), induit un flot de données perpétuellement changeant et souvent très volumineux qui remet en question l'algorithmique classique consistant à lire préalablement la donnée sur laquelle travailler. Ce qu'on pourrait appeler l'*algorithmique incrémentale* est présentement une branche de la discipline en plein développement.

Une science, l'autre pas

Cette perspective centrée sur les périphériques n'inclut pas la dénommée « intelligence artificielle », ce qui appelle plusieurs remarques :
– ce domaine repose essentiellement sur deux piliers, le modèle de réseau neuronal, emprunté à la biologie, et la notion de base de connaissances, empruntée à une branche de l'informatique évoquée précédemment, les *bases de données* ; ces concepts anciens donnent de meilleurs résultats aujourd'hui grâce à l'accélération des processeurs et à l'agrandissement des mémoires, sans avoir nécessité de franchissement d'un nouvel obstacle épistémologique ;
– comme déjà signalé, la science étudie des phénomènes, non des objets manufacturés ; à ce titre, l'IA n'est pas une science mais plutôt un ensemble de technologies, issues hier de la mécanique, aujourd'hui de l'informatique, demain peut-être de la biologie ;
– l'informatique a d'abord été vue comme une branche des mathématiques ou de la physique, et a fini par s'en dissocier, étant incompatible avec les épistémologies respectives de ces deux disciplines ; l'IA prendra sans doute son indépendance de l'informatique, pour entretenir avec elle des rapports assez semblables à ceux de la médecine avec la biologie.

Bachelard suggérait que les obstacles épistémologiques reposaient souvent sur un vocabulaire inadapté ; sur ce plan, l'IA en fournit un florilège intéressant :

– la locution « intelligence artificielle », traduite mot à mot de l'anglais alors que le mot français « intelligence » se traduit par « *cleverness* » (contre-performance pour une activité incluant la traduction automatique) est doublement trompeur ; parler de « savoir numérique » serait plus juste ;
– la locution « apprentissage profond », moins employée que son original anglais « *deep learning* » témoigne d'un anthropocentrisme discutable, parler de « mimétisme massif » décrirait mieux la technologie sous-jacente, et il est significatif que ses échecs soient nommés « *over-fitting* », le mot « plagiat », qui devrait s'imposer, étant évidemment moins discret.

Avec de telles approximations, il n'est pas étonnant que les adversaires de la science l'accusent de « ne pas penser ». Il faut renoncer à l'inflation du vocabulaire scientifique dans l'espoir d'obtenir plus de crédits, et revenir aux étymologies grecques, latines ou arabes pour forger un vocabulaire précis.

Jean Rostand dans sa passionnante histoire de la biologie, expose que la question centrale de la biologie est « qu'est-ce que la vie ? », qui reste toujours sans réponse mais permet de faire avancer la discipline. On pourrait reprendre la formule pour physique & matière, psychologie & individu, et bien d'autres.

Et l'informatique ?

– Étudierait-elle l'intelligence ? Non, on peut prendre comme contre-exemple une création musicale par ou avec un ordinateur, dont on attend une séduction sonore plus qu'une démonstration d'intelligence. Laissons cette étude à la psychologie génétique, ou peut-être aux mathématiques.

– Au vu de la variété des architectures matérielles et logicielles, c'est semble-t-il une question plus générale et plus humaniste : « qu'est-ce que la pensée ? »

Hommage à Jean-Pierre Archambault

Jacques Baudé
Président d'honneur de l'EPI

Jean-Pierre nous a quitté le 23 février dernier, il nous manque déjà !

Nous nous sommes rencontrés pour la première fois au début de la décennie 1980. Ce fut le début d'une longue amitié de près d'un demi-siècle. Nous avions tout pour nous entendre. Nous militions tous deux au SNES et à l'EPI[1], nous pratiquions tous deux l'approche collective des problèmes, nous faisions tous deux une analyse de gauche des problèmes de l'Éducation nationale. Combien de fois l'ai-je entendu dire que l'éducation n'est pas une dépense mais un investissement ! A fortiori quand on parle de l'informatique…

Mais comment rendre hommage à Jean-Pierre sans évoquer d'abord ses multiples activités dès le début des années 1970.

Après une dizaine d'années d'enseignement des mathématiques au cours desquelles il perçoit l'intérêt des logiciels pour sa discipline, il suit, en 1981–82, une formation approfondie d'un an (dite « formation lourde ») à l'informatique pédagogique, assurée par l'équipe de Jacques Arsac à l'ENS.

1. EPI : association « Enseignement public & informatique ».

DOI:10.48556/SIF.1024.26.115

Il est ensuite recruté comme formateur dans l'académie de Créteil, ce qui lui permet de participer activement au pilotage du développement des technologies de l'information et de la communication (TIC) dans l'académie : organisation du volet formation du plan « Informatique pour tous »[2], mise en œuvre de la télématique scolaire et des réseaux locaux, formation des enseignants.

En 1996, il rejoint la direction de l'ingénierie éducative du CNDP. En 1999, il est chargé de la mission de « veille technologique » dès sa création. Ce service du CNDP-CRDP de Paris s'intéresse aux usages pédagogiques de l'ordinateur. Les logiciels libres constituent l'un des « chantiers » qui lui sont confiés par la direction générale. Ce chantier s'inscrit dans le cadre de l'accord signé en octobre 1998 par le ministère de l'Éducation nationale et l'AFUL[3], accord qui dit en substance que « les logiciels libres constituent une solution alternative de qualité, à moindre coût, pour les établissements scolaires, dans une perspective de pluralisme technologique[4] ».

Dans une longue interview[5] parue en septembre 2004 dans le magazine de l'ASTI[6], STIC-Hebdo nº 26, il commente les espoirs mais aussi les difficultés que rencontre le déploiement du logiciel libre. Je cite sa conclusion :

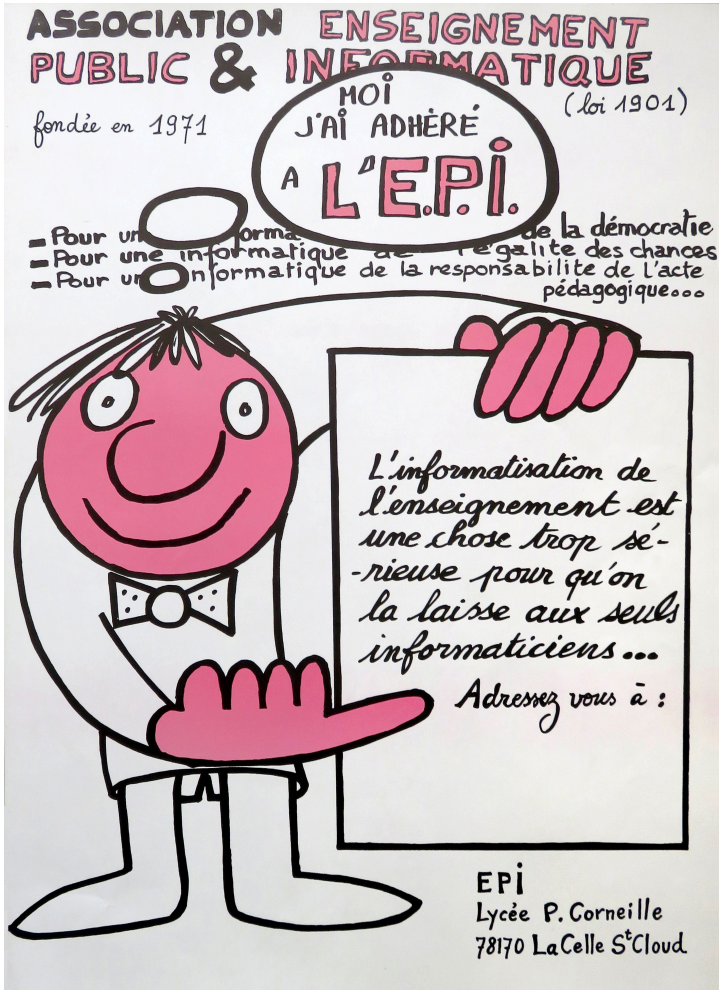
> *« Nous sommes dans une période de transition, avec des rapports de force incertains, mais qui évoluent en faveur du libre. Cette période est passionnante car les enjeux éducatifs du libre sont forts. Et les questions posées, des questions de société. Avec, en toile de fond, le pluralisme, la diversité, l'accès libre pour tous à la connaissance, à la culture. »*

Quand, en 2007, il accepte de prendre la présidence de l'EPI, il me demande de l'aider. Il a dans la tête depuis un moment déjà l'idée de promouvoir à nouveau un enseignement de la science et technologie informatique dans le secondaire en complément de l'utilisation de « l'outil » informatique dans les disciplines, et en liaison avec le logiciel libre. C'est une demande récurrente de l'EPI qui, faute de relai, a vécu comme un traumatisme la double suppression de l'option informatique des lycées en 1992 et 1998[7], et qui pratique le libre depuis la décennie 70.

Il reste alors à convaincre le plus largement possible !

---

2. Voir ce bulletin, nº 5, *Le plan « Informatique pour tous »*, DOI:10.48556/SIF.1024.5.95, et également nº 6 : *Le plan Informatique pour tous dans l'académie de Créteil : une enquête d'évaluation*, DOI:10.48556/SIF.1024.6.97.
3. AFUL : association francophone des utilisateurs de logiciels libres, `https://aful.org/`.
4. Le libre dans l'Éducation nationale, Jean-Pierre Archambault, Framabook, `https://archives.framabook.org/docs/Histcultlib/Complements/FBook_JPArchambault_CCBy.pdf`.
5. Cinq questions à Jean-Pierre Archambault, CNDP-CRDP de Paris, coordonnateur du pôle de compétences logiciels libres du SCEREN. `https://www.epi.asso.fr/revue/articles/a0409b.htm`.
6. ASTI : fédération française des associations des sciences et technologies de l'information.
7. L'option informatique (trois parties) : `https://edutice.hal.science/edutice-00564559v1`.



Affiche[8] créée en 1980 par André Poly.

8. L'adresse sur l'affiche était celle de l'EPI avant l'achat du local de la rue du Jura (à Paris) en 1986.

Pour l'EPI, il est fondamental que l'informatique soit enseignée dès le collège et le lycée, mais il est également essentiel que les élèves y soient sensibilisés dès l'école primaire. Jean-Pierre défend partout l'idée que « l'enseignement de l'informatique et son utilisation dans les disciplines sont deux démarches complémentaires » et milite pour la création de formations solides pour les enseignants.

En 2005 a lieu un débat à l'Académie des sciences sur la thématique générale « *L'enseignement de l'informatique de la maternelle à la terminale* ». Y interviennent Gérard Berry, Gilles Dowek et Maurice Nivat. Ce dernier, dans son intervention, propose d'introduire au lycée « *un véritable enseignement d'informatique formant aux notions essentielles d'algorithme et de programme* », proposition à laquelle Gérard Berry et Gilles Dowek adhèrent immédiatement.

J'ai déjà eu l'occasion d'écrire[9] que ce débat a servi de déclic pour les responsables de l'EPI qui vont reprendre leur bâton de pèlerin.

En janvier 2007, l'association adresse une lettre ouverte aux candidates et candidats à la présidence de la République. En février 2007, l'April[10] et l'EPI interrogent les candidats à l'élection présidentielle. Nicolas Sarkozy, futur président de la République, dans sa réponse évoque « *la refonte des programmes éducatifs consacrés à l'informatique [actuellement] trop centrés sur la pratique* ».

Le 26 août 2007, l'EPI lui adresse une demande d'audience, et Jean-Pierre et moi sommes reçus à l'Élysée par le conseiller technique du président pour l'éducation, puis le 12 décembre 2007, au MEN, par Mark Sherringham, inspecteur général, conseiller au cabinet du ministre de l'Éducation nationale, Xavier Darcos.

Ainsi, à partir de 2005, nous entrons dans une période où, grâce à des interventions convergentes d'organisations et de personnalités — l'EPI n'est plus seule ! —, une certaine prise de conscience de l'importance d'un enseignement de l'informatique regagne lentement l'esprit des responsables… Par leur rôle important, les acteurs principaux sont l'Académie des sciences[11], le conseil national du numérique, Pascaline, l'INRIA, l'ASTI, les parents d'élèves de la PEEP, les inspecteurs généraux Robert Cabane et Laurent Chéno, et de nombreuses personnalités du monde informatique… J'en oublie, mais certainement pas la SIF créée en 2012 !

9. Éléments pour une histoire de l'enseignement de l'informatique dans l'enseignement général (école, collège, lycée) en France (1970-2017). Un développement chaotique et inachevé. Jacques Baudé, 2e partie (2000–2017), `https://www.epi.asso.fr/revue/histo/h17_jb-hist-info-2.htm`.

10. April est une association de promotion et de défense du logiciel libre dans l'espace francophone.

11. Jean-Pierre Archambault fut membre du groupe de travail de l'Académie des sciences qui publia en 2013 un rapport « *L'enseignement de l'informatique en France — Il est urgent de ne plus attendre* », plaidant pour un enseignement structuré de l'informatique dans le système éducatif français. `https://www.epi.asso.fr/revue/docu/d1305a.htm`.

Le rôle de Jean-Pierre fut important, il ne ménageait pas ses efforts et avait l'art de convaincre. Yves Bertrand, dans son éditorial du numéro 25 de 1024, le qualifie d'influenceur ; à l'EPI nous préférions le terme de « catalyseur »… Jean-Pierre explique sans relâche :

> *« Si les sciences physiques sont devenues une discipline scolaire il y a plus d'un siècle, c'est parce qu'elles sous-tendent les productions de la société industrielle. D'une manière analogue, l'informatique, qui est sous-jacente aux réalisations de la société numérique, doit devenir discipline scolaire de culture générale pour tous les élèves ».*

Je ne reviens pas ici sur les actions collectives qui ont engagé l 'ASTI, SPECIF, le groupe ITIC[12], puis la SIF… et qui ont abouti à des avancées importantes, dont la création d'un CAPES NSI et d'une agrégation d'informatique. Mais Jean-Pierre avait pleinement conscience des difficultés que rencontre la spécialité NSI telles qu'elles sont clairement exposées dans 1024 nº 25. Son analyse était que la réforme Blanquer n'est pas la solution et qu'elle n'a fait qu'aggraver la situation de l'enseignement scientifique au lycée au moment où des défis considérables se posent dans notre pays et en l'Europe (IA, cybersécurité, ordinateur quantique, addiction aux smartphones…).

L'April, le site Linuxfr.org[13], ainsi qu'Alexis Kauffmann — son collègue et ami de longue date qui « *marche dans ses pas aujourd'hui* » — et bien d'autres encore lui ont rendu hommage dès la nouvelle de son décès. Le compte « Le Libre éducatif » (compte de la Direction du numérique pour l'éducation du ministère de l'Éducation nationale dédié aux logiciels et ressources éducatives libres) a salué sa mémoire sur X :

> *« En tant que président de l'EPI (enseignement public et informatique) il a été un moteur pour l'intégration de l'informatique dans les programmes scolaires […] Sa passion pour le logiciel libre et son implication dans le développement de ressources éducatives libres ont inspiré de nombreux enseignants. Son travail a permis de démocratiser l'accès à l'informatique et de promouvoir une approche pédagogique innovante. »*

Jean-Pierre a participé à de nombreux débats, colloques, séminaires. Il a écrit de très nombreux articles — sur les logiciels libres, les ressources pédagogiques libres, les modèles économiques et la société, l'informatique pédagogique (expression comprenant, selon l'acception EPI, l'enseignement de l'informatique) — articles riches d'enseignement, qui sont en ligne sur le

12. Historique du groupe ITIC, Jean-Pierre Archambault, Jacques Baudé ; EpiNet nº 204, avril 2018. https://www.epi.asso.fr/revue/histo/h07_groupe-itic_jb-jpa-18.htm. Première réunion fondatrice : https://edutice.hal.science/edutice-00277825/file/asti-itic-cr_0709.htm.
13. https://linuxfr.org/.

site EPI[14], sur l'archive Edutice[15] et sur des sites de presse et sites amis ; et il attachait un soin extrême à rédiger les éditoriaux mensuels d'EpiNet, je peux en témoigner. Je renvoie à son abondante bibliographie et à sa webographie[16].

Dans son ultime combat contre la maladie, il s'inquiétait de l'avenir de l'EPI et de l'enseignement de l'informatique mais se réjouissait de l'existence de la société Informatique de France. Il reste tant à faire…

Au revoir Jean-Pierre et merci !

Jacques Baudé
Septembre 2025

14. https://www.epi.asso.fr/.
15. https://edutice.hal.science/.
16. https://www.epi.asso.fr/biblio/jpa_bibliographie.htm.

Le temps d'un au revoir[1]

La SIF

Le décès prématuré de Gilles Dowek, le 21 juillet 2025, a touché et touche encore toutes les personnes qui ont eu la chance de le croiser. Le texte qui suit a été écrit par la SIF pour lui rendre hommage. Ce n'est qu'une voix parmi d'autres, le texte ne se veut pas exhaustif et ne prétend pas aborder tous les aspects d'une vie riche et multifacette. D'autres hommages viendront dans les mois à venir, notamment l'édition 2026 du congrès de la SIF, qui sera dédié à sa mémoire.

Portrait de Gilles Dowek – © Sébastien Dolidon.

1. *Le temps des algorithmes*, avec son compère Serge Abiteboul, Le Pommier, 2017.

DOI:10.48556/SIF.1024.26.121

Ni périphrase, ni circonlocution : Gilles était attaché à l'emploi des mots justes. Et à quoi bon user d'une quelconque figure de style ? Aucune ne saurait atténuer l'affliction dans laquelle nous plonge le départ prématuré de notre collègue, compagnon de route et pour certains, ami.

Sous un faux air d'Alain Chabat, et doté d'un vrai charisme de savant humaniste à la fois drôle, cultivé et profond, Gilles est l'une des bonnes fées qui se sont penchées sur le berceau de la Société informatique de France à sa naissance. Il lui a transmis ses premiers dons : quatre piliers scientifiques[3] pour ancrer la discipline informatique, et une exigence éthique[4] pour la mettre au service de l'être humain et non l'asservir. Tout au long de ces années, il a su être un guide précieux pour accompagner sa croissance. Jamais dogmatique, toujours ouvert au dialogue, il était prompt à initier et alimenter les réflexions, notamment au sein du conseil scientifique. Héritier des Lumières, il était aussi à l'aise à discourir sur les sciences que sur les humanités. Il était particulièrement attaché à offrir aux jeunes la possibilité de se développer simultanément dans ces deux domaines et préconisait de « faire de l'enseignement des sciences une grande cause nationale »[5] en lui redonnant une place équivalente à celle des humanités dans le tronc commun du lycée. Passeur de sciences informatiques, il a œuvré avec énergie pour qu'elles deviennent accessibles à toutes et tous, et fassent partie intégrante de la formation des citoyennes et citoyens de notre temps[6].

Au séminaire des doctorantes et doctorants en informatique de la SIF en juillet 2019 – © Young-Ah Kim.

Sans jamais regarder quiconque de haut, il nous aidait à prendre de la hauteur. Il n'en était pas pour autant déconnecté des réalités de terrain, savait poser des gestes simples et encourageait les métamorphoses[7] de la pensée en actes signifiants.

2. *Ce dont on ne peut parler, il faut l'écrire*, Le Pommier, 2019.
3. *Les quatre concepts de l'informatique*, https://edutice.hal.science/edutice-00676169/fr/.
4. *Un objet informatique a-t-il une éthique ?*, *in* É. Germain, C. Kirchner, C. Tessier, « Une éthique du numérique : pour quoi faire », Puf, 2022.
5. https://archive.socinfo.fr/wp-content/uploads/2022/01/EnseignementDesSciences-GrandeCauseNationale.pdf.
6. *L'enseignement de l'informatique — Il est urgent de ne plus attendre*, rapport de l'Académie des sciences, https://www.academie-sciences.fr/pdf/rapport/rads_0513.pdf.
7. *Les métamorphoses du calcul — Une étonnante histoire de mathématiques*, Le Pommier, 2007.

Très investi à la SIF, il l'était aussi ailleurs : membre actif de plusieurs comités d'éthique, cheville ouvrière du conseil supérieur des programmes… il savait assumer des engagements aussi bien professionnels que personnels.

Reconnu par ses pairs, il avait notamment été récompensé coup sur coup par l'Académie des sciences, en 2023 pour ses apports scientifiques, en 2024 pour son travail en histoire des sciences et épistémologie. Les paris allaient bon train pour savoir quelle récompense l'Académie lui attribuerait en 2025.

Nous le savions malade, il en parlait librement, sans pathos, mais il avait déjà largement déjoué les pronostics et nous voulions croire qu'il en serait de même encore longtemps.

Hélas, aujourd'hui, Gilles Dowek s'est fait hacker[8] ! Nous perdons un scientifique, un médiateur, un philosophe, un homme qui aura inspiré nombre d'entre nous par ses qualités scientifiques et humaines. Il laisse un riche héritage, écrit et filmé, qui traite de questions intemporelles et qui n'a pas fini de nous inciter à réfléchir. Il nous lègue aussi un regard optimiste dénué de tout angélisme, une invitation à poursuivre le développement de notre intelligence humaine les yeux grand ouverts sur le monde.

Avec Michel Serres, au congrès de la SIF en 2018.

8. *Qui a hacké Garoutzia ?*, avec Serge Abiteboul et Laurence Devillers, C&F éditions, 2023.

Et parce que sa voix ne s'est pas éteinte avec lui, laissons-lui le mot de la fin : la retranscription d'un extrait d'une intervention réalisée à l'occasion des rencontres philosophiques Michel Serres en 2023 :

« Je ne cherche pas spécialement à vous rassurer […], j'essaie juste de comprendre le monde dans lequel nous vivons et je crois qu'il n'est pas possible de le comprendre si la première question que l'on se pose est : « Est-ce que c'est bien ou est-ce que c'est mal ? », c'est-à-dire que cette question va bien sûr venir à un moment, mais les premières questions doivent être : qu'est-ce que c'est ? comment ça marche ? d'où ça vient ? Et c'est seulement dans un second temps qu'on peut se poser la question de savoir si c'est bien ou si c'est mal […] »[9].

9. *Comment l'IA peut changer le monde*, retranscription (légèrement arrondie) de quelques minutes de son intervention (à 17 minutes et 16 secondes), https://www.youtube.com/watch?v=Bx4KR50TTXo.

Hommage à Gilles Dowek

Jacques Baudé
Président d'honneur de l'EPI

C'est avec beaucoup de tristesse que j'ai appris le décès de Gilles. C'est la disparition d'un collègue brillant et aussi d'un ami comme l'était Jean-Pierre Archambault. Cela fait beaucoup en si peu de temps ! Je tiens à lui rendre hommage en mon nom et au nom de l'EPI[1] récemment dissoute.

De gauche à droite : Maurice Nivat, Gérard Berry, Jean-Pierre Archambault, Gilles Dowek.

D'autres que moi diront son parcours universitaire, ses apports scientifiques, historiques, éthiques et philosophiques, aussi je ne traiterai ici que du rôle important qu'il a joué dans la promotion de l'enseignement précoce de la

1. EPI : association « Enseignement public & informatique ».

DOI:10.48556/SIF.1024.26.125

science informatique. Les comptes rendus des séances du groupe « Informatique et TIC » en ligne sur le site de l'EPI m'ont servi de référence[2].

C'est au cours de ces réunions de travail — cogérées avec Jean-Pierre pendant plusieurs années — que j'ai pu apprécier chez Gilles, ses compétences, son humanité et son humour. C'était un plaisir de suivre ses interventions, je ne m'en lassais pas.

Dès 2007, il participe à ce groupe[3] de l'ASTI[4] (plus spécialement au sous-groupe « lycée » avec Jean-Pierre Archambault, Gérard Blanchet et moi). Il s'agit alors de répondre à la commande sur le programme d'informatique au lycée qu'a faite Mark Sherringham (conseiller du ministre Xavier Darcos) — que l'EPI vient de rencontrer le 12 décembre, à la suite de notre audience à l'Élysée.

En avril 2008, au séminaire du groupe, qui se déroule à Marseille dans le cadre des Rencontres de l'Orme, Gilles présente les grandes lignes de ce programme « lycée » ainsi que les principes qui nous ont guidés dans son élaboration. Extrait du compte-rendu signé Jean-Pierre Archambault, alors président de l'EPI depuis un an :

> *« Dans la table ronde de l'après-midi, Maurice Nivat a rappelé ce qu'était la science informatique (dont les concepts sont "cachés" dans les outils utilisés), à savoir l'algorithmique, la programmation et la théorie de l'information. Il a souligné qu'un programme pour le lycée ne pouvait qu'être centré sur ce qu'est la science informatique. Monique Grandbastien a indiqué qu'il est parfaitement possible d'enseigner l'informatique au lycée : tout simplement parce que ce fut le cas dans les années 80 dans une option d'enseignement général. Cet enseignement était une réussite, avant que le ministère le supprime au début des années 90 ! Gilles Dowek a de nouveau présenté le programme. L'idée de faire passer les apprentissages par les utilisations dans les autres disciplines est peut-être "séduisante", mais la réalité prouve que cela ne marche pas : c'est ce que montre notamment le constat que l'on peut faire à l'entrée des élèves dans les grandes écoles. Éric Bruillard a indiqué, à partir des résultats de recherches menées, que les élèves de collège ne maîtrisaient nullement le tableur après une initiation limitée et discontinue dans le temps. Il a également montré qu'un enseignement de l'informatique en tant que telle était incontournable pour dispenser une culture informatique . »*

Le 24 septembre 2008, Jean-Pierre et Gilles sont reçus par le recteur Jean-Paul de Gaudemar, qui pilote la mission sur la réforme du lycée, et Érick Roser,

2. https://www.epi.asso.fr/blocnote/blocsom.htm#itic-epi-sif.
3. Historique du groupe ITIC-EPI-ASTI, puis ITIC-EPI-SPECIF et ITIC-EPI-SIF : https://www.epi.asso.fr/revue/histo/h07_groupe-itic_jb-jpa-18.htm.
4. ASTI : fédération des associations françaises en sciences et technologies de l'information.

IGEN de mathématiques. L'objet de la rencontre porte à nouveau sur l'opportunité de créer un enseignement de l'informatique au lycée.

Ils communiquent la proposition de programme élaborée par le groupe ITIC. La rencontre donne lieu à « des échanges approfondis et constructifs ». Une discipline de culture générale scientifique et technique aurait pour finalité de former les élèves au monde numérique.

Jean-Paul de Gaudemar demande qu'on lui fasse des propositions sur ce que pourraient être un module de présentation de la discipline en seconde (enseignement semestriel de 54 heures) et des modules en première et terminale, ceux-ci étant soit thématiques soit « d'initiation et d'approfondissement ».

S'oriente-t-on vers un changement de paradigme ?

Au Salon Educatice 2008 , Gilles participe, avec Gérard Berry, Marie-Christine Milot (MEN, sous-Direction des technologies de l'information et de la communication pour l'Éducation) et Bruno Devauchelle (Café pédagogique), à la conférence organisée par l'EPI : « *Comment donner une culture générale informatique à tous les élèves ?* »[5].

Gilles précise d'emblée qu'il faut sortir du débat « *informatique en tant qu'objet ou en tant qu'outil* ». Quand on part des usages en informatique, il y a un risque de redondance avec ce que savent les jeunes. Et apprendre à envoyer un mail n'est pas suffisant ! Je le cite :

> *« Les concepts informatiques existent depuis très longtemps déjà, ce sont les mêmes depuis les années trente : algorithme, machine et réseau, langage, information. Il s'agit donc des connaissances stables.*
>
> *Autour de quelles notions l'informatique est-elle structurée ?*
>
> *1re notion : le concept d'algorithme (naissance 3500 avant JC) qui a amené les mathématiques et peut s'appliquer à tous les métiers ;*
>
> *2e notion : le langage (lien avec de nombreuses disciplines : langues, français, philosophie…) ; variété des langages de programmation qui ont en commun une exigence de rigueur (si un point-virgule manque, l'ensemble du message est "cassé") ;*
>
> *3e notion : la machine, les ordinateurs, les réseaux ; une machine est incarnée dans des objets physiques et il faut l'"ouvrir" pour connaître ses caractéristiques ;*
>
> *4e notion : l'information ; la notion est centrale. »*

En réponse à la question de savoir quoi et quand enseigner en termes de connaissances numériques, Gilles propose un schéma en trois temps :

> *« 1re étape : à l'école élémentaire et au collège, familiarisation avec les outils, pas simplement vivre avec, mais savoir se servir d'un logiciel de traitement de texte, d'une messagerie, de l'Internet ;*

5. https://www.epi.asso.fr/revue/docu/d0901a.htm.

2e étape : le lycée doit aller plus loin, donner les explications que l'on cherche, structurer les savoirs, apprendre à écrire soi-même un programme ;

3e étape : l'université. L'informatique est une science avec des choses vraies et des choses fausses, très vaste, avec de très nombreuses spécialités (théorie des réseaux, des bases de données…).

Il s'agit d'une formation scientifique de connaissances et d'un cursus qui s'intéresse à tous car tous vivent dans "la société numérique".

Informatique pour tous ? Oui, comme pour l'enseignement de la biologie, qui permet de comprendre le vivant sans être biologiste, même si de façon secondaire cela permet aussi de former des biologistes. L'objectif est de fournir les clefs pour comprendre le monde, c'est-à-dire donner la liberté d'agir, d'être sujet plutôt qu'objet. »

Gilles ne ménage pas sa peine et participe activement aux rencontres tous azimuts car il ne s'agit pas seulement de s'adresser au MEN ! Ainsi, le 28 avril 2009, Jean-Pierre et lui sont reçus au secrétariat d'État de la prospective et du développement de l'économie numérique où ils présentent une fois de plus la réflexion et l'action du groupe « Informatique et TIC » en faveur d'une discipline informatique en tant que telle au lycée. Ils rappellent que la réforme du lycée, reportée à la rentrée 2010, comporte la création d'un module « Informatique et société numérique ».

Mêmes démarches auprès de la DGESCO, de la mission Fourgous, du cabinet du MEN , de Matignon… mais il me faut abréger. Tout cela se trouve dans les archives du site EPI [6].

Retenons cette démarche collective que Gilles appréciait tout spécialement : le 22 mars 2010, au nom de l'EPI et du groupe ITIC,[7] Jean-Pierre Archambault, président de l'EPI, Gérard Berry, membre de l'Académie des sciences et professeur au collège de France, Gilles Dowek, professeur d'informatique à l'École polytechnique, et Maurice Nivat, membre correspondant de l'Académie des sciences, sont reçus au cabinet du ministre de l'Éducation nationale (Luc Chatel) par Érick Roser, Conseiller pour les affaires pédagogiques, et Benoît Labrousse, conseiller technique (nouvelles technologies, éditeurs, multimédia).

L'audience porte essentiellement sur la formation des professeurs (deuxième fil d'Ariane pour l'EPI et l'ASTI !) qui assureront l'enseignement de spécialité optionnel « Informatique et sciences du numérique » en terminale S à la rentrée 2012. L'audience est l'occasion de rappeler que l'EPI et le groupe ITIC se sont félicités de la création de cet enseignement de spécialité, première phase du processus de développement de l'enseignement de l'informatique au lycée. Les deux organisations présentent leurs propositions (archives EPI).

6. https://www.epi.asso.fr/blocnote/blocsom.htm#090928.

7. Groupe ITIC : enseignement de l'informatique et des technologies de l'information et de la communication.

En mars 2011, Gilles intervient à l'ENS de Lyon au séminaire organisé par la direction générale de l'enseignement scolaire et l'inspection générale de l'Éducation nationale à l'intention des IPR-IA. Il s'agit pour lui de sensibiliser l'inspection à l'impérieuse nécessité de former les enseignants.

Peu après, accompagné par Serge Abiteboul et Jean-Pierre Archambault, il rencontre une délégation de l'inspection générale, conduite pas Jean-Louis Durpaire et Michel Pérez. sur la nécessité d'un enseignement de l'informatique pour tous les élèves dès l'école primaire et ils traitent à nouveau l'importante question de la formation des enseignants.

Au salon Educatec-Educatice 2013 est programmée une conférence sur le fameux rapport de l'Académie des sciences «*L'enseignement de l'informatique en France - Il est urgent de ne plus attendre*»[8]. Elle réunit une fois de plus Jean-Pierre Archambault, Serge Abiteboul et Gilles Dowek qui avaient participé à l'élaboration de ce rapport.

En mai 2013, Serge Abiteboul, Jean-Pierre Archambault (EPI), Gérard Berry, Colin de la Higuera (SIF), Gilles Dowek et Maurice Nivat s'adressent à Pierre Bel, président du Sénat, «*Pour une discipline informatique dans l'enseignement secondaire. Enjeu majeur pour notre pays. Demande d'intervention lors du débat sur la refondation de l'École*» .

Le groupe ne se limite pas au lycée. Il adresse au conseil supérieur des programmes une «*Proposition d'orientations générales pour un programme d'informatique à l'école primaire* » et, en janvier 2014, il publie et fait connaître «*Esquisse d'un programme d'informatique pour le collège*».

À l'initiative de Gilles, la 22e réunion du groupe ITIC-SIF-EPI a lieu le 20 juin 2014 à l'INRIA et il tient à en cosigner le compte rendu avec Jean-Pierre et moi. Il aimait souligner la qualité des travaux de ce groupe qui ont inspiré bien des démarches auprès des autorités administratives.

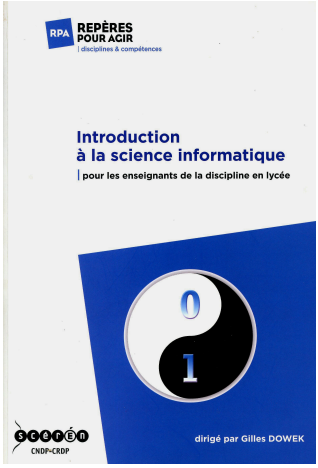
Ne ménageant pas ses efforts, Gilles représente le groupe au séminaire sur l'enseignement de l'informatique aux Pays-Bas, en septembre 2014, où il fait deux interventions l'une sur la situation de la France, l'autre sur la construction d'un programme d'informatique. Les «rapports par pays» ont mis en avant beaucoup de similarités. La plupart des pays ont déjà des professeurs d'informatique dans les écoles, collèges et lycées, et sont en train de revoir leurs curricula à la hausse.

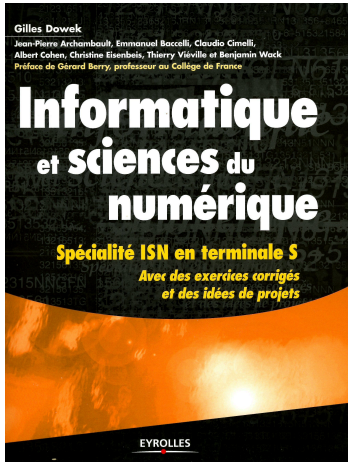
Il s'agit de convaincre au-delà des cercles déjà convaincus… En mai 2015, Gilles intervient au congrès national de la PEEP avec Jean Pierre Archambault et Colin de La Higuera (alors président de la SIF) dans la table ronde «l'École au cœur du numérique». La même année, il participe à la rédaction d'un texte commun avec le CIGREF sur l'enseignement de l'informatique et son importance pour les entreprises.

8. https://www.epi.asso.fr/revue/docu/d1306a.htm.

La 31e réunion du groupe ITIC-EPI-SIF, qui a lieu à l'ENS Cachan le 16 octobre 2019, est consacrée à la rentrée dans les lycées, du point de vue de ses aspects informatiques dans le cadre de la réforme du lycée et est suivie de la première session du séminaire franco-argentin sur l'enseignement de l'informatique (coorganisé par Gilles). Cette rentrée voit les créations de l'enseignement de spécialité NSI en première et de SNT pour tous les élèves de seconde…
Pour terminer, j'évoque les deux ouvrages écrits à l'initiative de Gilles.

En 2011, il dirige un ouvrage «*Introduction à la science informatique, pour les enseignants de la discipline en lycée* », édité par le CRDP de Paris avec le soutien de l'ASTI et de l'EPI et préfacé par Gérard Berry. Cet ouvrage de 376 pages est alors sans équivalent dans la discipline à ce niveau d'enseignement et est sous licence Creative commons «paternité, pas d'utilisation commerciale, pas de modification». On y note la présence de Jean-Pierre Archambault…





Il récidive l'année suivante en dirigeant un manuel collectif de 300 pages «*Informatique et sciences du numérique — Spécialité ISN en terminale S*» publié chez Eyrolles avec le concours de l'EPI, la SIF et INRIA, et à nouveau préfacé par Gérard Berry. L'objectif de ce cours est d'introduire les quatre concepts de machine, d'information, d'algorithme et de langage qui sont au cœur de l'informatique et de montrer comment ils fonctionnent ensemble. Ce manuel est toujours en ligne à cette URL[9] et continue à être utilisé par les élèves et enseignants de NSI.

9. https://wiki.inria.fr/sciencinfolycee/Informatique_et_Sciences_du_Numérique-_Spécialité_ISN_en_Terminale_S.

Je m'arrête là, ayant bien conscience que ce petit hommage est incomplet tant les activités de Gilles étaient multiples dans les domaines qui nous concernaient.

Par son action au sein d'équipes — car il aimait les démarches collectives — il a participé activement aux avancées significatives qui ont été faites : ISN, SNT, NSI, formation des maîtres, CAPES et agrégation, …. (pour ses compétences informatiques et pédagogiques, Gilles avait été nommé membre du conseil supérieur des programmes).

Il reste à confirmer ces avancées, les compléter et les amplifier tant les besoins sont importants.

Il reste à répondre aux défis de l'IA et de l'impact des smartphones…

Que l'exemple de Gilles continue à nous inspirer !

Jacques Baudé
Ex-membre du Comité de suivi de l'option informatique (1980–1990)
Septembre 2025

Mémoire vive – 2007 : entretien avec Gilles Dowek

Pierre Lescanne
Professeur émérite, ENS de Lyon

En décembre 2007, Pierre Lescanne réalisait – pour le numéro 58 du bulletin de SPECIF – une interview de Gilles Dowek à l'occasion de la publication de son livre, «Les métamorphoses du calcul» (ce livre venait de recevoir le grand prix de philosophie de l'Académie française). Nous reproduisons ici, en hommage à Gilles Dowek, cette interview qui, nous semble-t-il, illustre bien un aspect de sa pensée sur l'informatique, la mathématique… et les autres sciences.

SPECIF (SP). *Ton livre «Les métamorphoses du calcul» vient d'obtenir le grand prix de philosophie de l'Académie française. Aurais-tu abandonné l'informatique pour devenir un philosophe ?*

Gilles Dowek (GD). Pas du tout. La philosophie des sciences s'est souvent construite dans un dialogue entre les philosophes et les scientifiques, qui apportent des matériaux complémentaires à cette construction. Un informaticien peut donc écrire un livre de philosophie des sciences sans, pour cela, avoir besoin de changer de métier.

SP. *Le sous-titre de ton livre est «une étonnante histoire de mathématiques», s'agit-il aussi d'«une étonnante histoire de l'informatique» ?*

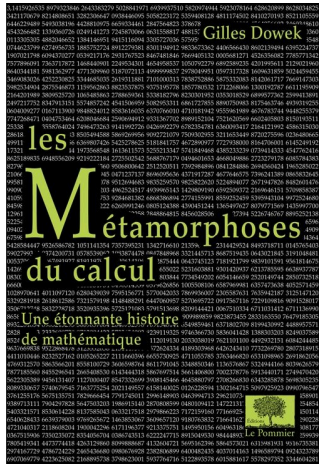
GD. En prenant comme sous-titre «une étonnante histoire de mathématiques», et non «des mathématiques», j'ai voulu insister sur le fait que l'histoire que je raconte concerne les mathématiques, mais qu'elle n'a pas l'ambition d'être toute l'histoire des mathématiques. De même, on peut dire qu'elle concerne l'informatique, mais elle n'a certainement pas l'ambition d'être toute l'histoire de l'informatique. Une des difficultés que nous avons,

DOI:10.48556/SIF.1024.26.133

aujourd'hui, pour faire émerger une philosophie de l'informatique est qu'il y a probablement une aussi grande diversité de problèmes et de méthodes en informatique que dans l'ensemble des sciences expérimentales. Il y a autant de différences entre la cryptologie, l'architecture des ordinateurs et la démonstration automatique, qu'entre la physique atomique, la biologie cellulaire et la chimie de la matière condensée. Il me semble donc nécessaire d'éviter les généralités et de penser localement.

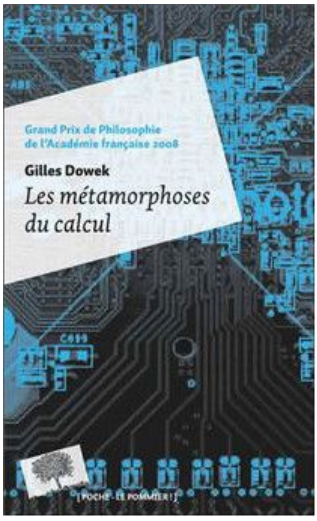
SP. *Ce livre, très accessible pour nous informaticiens, présente une thèse ou une vue du calcul dans une perspective historique et philosophique. Pourrais-tu nous la résumer en quelques mots ?*

GD. J'ai écrit ce livre parce que j'ai été frappé par le fait que des démarches relativement indépendantes, en logique, dans certaines branches de l'informatique et dans la pratique mathématique, aboutissaient à une conclusion similaire : celle de l'insuffisance de la méthode axiomatique. Cette méthode propose de construire des démonstrations en utilisant deux types d'ingrédients : des axiomes et des règles de déduction. Toutes ces démarches me semblent s'accorder sur l'idée qu'il faut un troisième type d'ingrédients : des règles de calcul. Un problème mathématique ne se résout pas uniquement en construisant un raisonnement, mais en articulant des étapes de raisonnement et des étapes de calcul. Je suis alors parti à la recherche des origines de cette idée, c'est-à-dire des moments de l'histoire où cette notion de calcul a joué un rôle important dans les mathématiques, ce qui m'a mené au développement des théories de la calculabilité et de la constructivité, puis aux algorithmes du calcul intégral, à la numération décimale, à l'algorithme d'Euclide et finalement aux mathématiques mésopotamiennes qui étaient purement algorithmiques, mais déjà très sophistiquées. Sur un plan plus philosophique qu'historique, cette idée qu'une théorie est constituée non seulement d'axiomes et de règles de déduction, mais aussi de règles de calcul, nous amène à repenser plusieurs questions, en particulier celle du rapport des mathématiques aux sciences expérimentales, celles de la place des instruments dans les sciences et celle de l'origine de la méthode axiomatique. Au delà des sciences, c'est peut-être la forme même de la connaissance qui est à repenser. On arrive ici à des questions philosophiques qui dépassent peut-être la stricte philosophie des sciences.



Édition originale.

SP. *Jan van Leuwen, vice-président d'Informatics Europe présentait les comptes rendus de son groupe de travail sur « Les effectifs étudiants et l'image de la discipline » en disant que l'une des faiblesses de l'informatique est son manque de support philosophique. Qu'en penses-tu ?*



Édition poche.

GD. L'image de notre discipline auprès des étudiants me semble assez brouillée. L'informatique semble être une théorie ésotérique, dans laquelle des machines écrivent sur des rubans, qui a des liens mystérieux avec des applications triviales, qui permettent de télécharger de la musique illégalement. Et être informaticien semble demander d'avoir subi un rite d'initiation avant l'âge de sept ans. Nous avons donc beaucoup de travail devant nous pour corriger cette image. Toutefois, nous devons aussi comprendre que la nature de l'informatique est difficile à comprendre, car notre discipline bouleverse la classification des sciences et également le rapport entre la science et la technique. Par exemple, nous revendiquons, avec raison, me semble-t-il, simultanément le statut de science à part entière et celui de technique ayant un impact fort sur le réel. Nous sommes à la fois astronomes et ingénieurs. Il n'est pas facile, quand on a été habitué à distinguer d'un côté la science désintéressée et de l'autre la technique efficace, d'abolir ces frontières pour penser la nouveauté de l'informatique.

SP. *Que penses-tu du statut de l'informatique dans le panorama de la science contemporaine ? Crois-tu que nos collègues des autres champs scientifiques connaissent les concepts qui la sous-tendent et que tu présentes dans ton livre ? En est-il de même des philosophes ?*

GD. La triste, mais inévitable, spécialisation de la recherche, fait que nous sommes nombreux à ignorer ce qui se fait dans les autres disciplines. Et comme l'informatique est enseignée peu et depuis peu, nombre de nos collègues n'ont même pas eu l'exposition à l'informatique qu'ils ont eue dans les autres sciences, quand ils étaient étudiants. Cependant, beaucoup d'entre eux ont perçu l'importance que l'outil informatique a pris dans leur propre pratique, comme outil de simulation et de traitement des données expérimentales. C'est à nous de les convaincre que l'informatique peut leur fournir non seulement des outils pour traiter leurs données, mais aussi des outils

pour penser les objets de leur science. En biologie, par exemple, l'informatique fournit non seulement un outil pour séquencer le génome, mais aussi un langage qui permet de décrire les processus cellulaires. Beaucoup de philosophes ont compris l'impact que les notions de calcul et de machine avaient sur leur discipline, en particulier sur les questions relatives à la nature de la pensée. Toutefois, la philosophie n'a pas comme unique but de penser la pensée. L'idée que j'essaie de défendre dans ce livre est que ces notions de calcul et de machine sont aussi utiles pour penser le monde.

SP. *Dans ton livre, tu nous brosses une histoire du calcul. Comment vois-tu son avenir ?*

GD. Voici la question à laquelle il vaudrait mieux répondre à l'encre sympathique, afin que la réponse s'efface avant qu'elle nous fasse éclater de rire. Mais aujourd'hui, il n'y à plus d'encre sympathique : tout ce qui est écrit l'est pour tous et pour une (courte) éternité. Nous avons maintenant plusieurs théorèmes qui ont été démontrés en utilisant une grande quantité de calculs, le théorème des quatre couleurs bien entendu, mais aussi l'inexistence d'un plan projectif d'ordre dix, le théorème de Hales… Nous pouvons donc commencer à faire des statistiques et constater que ces théorèmes ont un certain air de famille : ils appartiennent tous plus ou moins à la même branche de la géométrie. Une question qui se pose est celle de l'universalité de ces méthodes calculatoires. Allons-nous voir apparaître des démonstrations très calculatoires dans toutes les branches des mathématiques ou seulement dans certaines ? Et dans ce cas, lesquelles ? Une autre question qui prendra peut-être de l'importance dans le futur est celle de la nécessité de ce recours au calcul. Nous savons qu'il existe une démonstration calculatoire relativement courte du théorème des quatre couleurs, mais nous ne connaissons pas de démonstration axiomatique courte de ce théorème Cela signifie peut-être que nous n'avons pas assez cherché, mais peut-être aussi qu'il nen existe pas. Je crois que nous sommes encore loin d'avoir les outils pour démontrer que c'est le cas, mais il me semble difficile d'éviter de poser la question.

Émile 5.0

Marie-Fleur

30 décembre 2010

Léonie, 10 ans, marche d'un pas décidé aux côtés de son père. L'agence postale de Verzeille va bientôt fermer… mais pas avant qu'elle n'ait pu lui confier ses deux précieux paquets ! Deux lignes lancées dans l'espoir d'hameçonner l'avenir dont elle rêve…

Elle se presse. Un sourire radieux illumine son visage… Une fois sa mission accomplie, elle se précipitera, comme chaque soir, retrouver la vieille Sidonie, chez elle, dans sa bicoque biscornue. Cette curiosité d'un autre temps aujourd'hui enclavée dans un ensemble de constructions désespérément modernes marquait autrefois la frontière du village. Une frontière qui n'a jamais arrêté l'aïeule en son temps et ne stoppera certainement pas Léonie dans son élan. L'aînée et sa jeune « puînée », partagent la même énergie farouche, la même audace décomplexée. L'affection n'a pas d'âge et, entre ces deux-là, sœurs de cœur — mais pas de sang — une complicité unique s'est nouée au fil des années au rythme des récits et des rêves partagés…

Les dés sont maintenant jetés. Au moment précis où Léonie s'apprête à refermer la porte de l'agence postale, elle suspend un instant son geste ; le sifflet reconnaissable entre mille de la micheline en approche retentit deux fois, telle une promesse d'aventures résonnant dans l'air du soir.

10 septembre 2026

Armand Turin voyage dans le TGV qui le ramène chez lui, inconfortablement assis sur un tabouret de bar. Sur le comptoir devant lui, soigneusement empilés, l'attendent plusieurs quotidiens nationaux, sa pelote de fils d'Ariane pour le ramener à sa routine ordinaire. Le voilà prêt à découvrir, à rebours, ce qui a agité le monde ces quatre dernières semaines pendant sa

DOI:10.48556/SIF.1024.26.137

déconnexion annuelle. Il déplie les journaux pour en découvrir les gros titres. Son regard d'éditeur est aussitôt attiré par un titre qui, s'il ne fait pas la une, figure en bonne place : « Martial, destin d'une IA en lice pour le prix Goncourt ». Armand lève les yeux au ciel. Une IA pourvue d'un « destin », quelle aberration ! Une IA « Goncourt-able » qui plus est ! Que n'inventerait-on donc pas pour donner un coup de lustre 2.0 à ce prix ? Armand sent sourdre un agacement certain. Il se plonge dans la lecture de l'article. Les premières phrases lui arrachent un sourire désabusé : « Depuis sa sortie anticipée en plein cœur de l'été, le roman, "Marie" accapare la tête du palmarès des ventes. Du jamais vu pour un ouvrage entièrement écrit par une IA ». Quelle naïveté ! Comment croient-ils que sont aujourd'hui rédigés certains livres ? Depuis l'avènement des LLM — ces avatars numériques de madame IrmA, capables de prédire la suite la plus probable d'un texte selon un corpus précédemment appris — et leur popularisation, n'importe qui peut écrire un texte sans recourir à un prête-plume humain. Et pour peu que le livre soit signé d'une « personnalité », il caracole en tête des ventes. Armand poursuit sa lecture. L'irritation cède bientôt le pas à l'incrédulité. L'article cite l'interview d'une fameuse critique littéraire, qui ne peut être soupçonnée de complaisance ; or les qualificatifs qu'elle utilise pour décrire le roman « Marie » frôlent le dithyrambe. La Moire littéraire trois-en-une s'enthousiasme de la justesse des émotions ; elle souligne un choix de vocabulaire audacieux, mêlant expressions occitanes et françaises de différentes époques ; elle s'émerveille du rythme des phrases qui accompagne parfaitement la mélodie de l'histoire. Elle loue à la fois l'originalité de la forme littéraire et celle du récit. Armand est médusé : si un texte produit par une IA générative peut indéniablement présenter des qualités, il en est une dont il est par essence dépourvu : l'originalité.

Le train s'arrête. Armand attrape son sac à dos et descend prestement sur le quai. Il se faufile dans la marée humaine pour rejoindre le café où l'attend son associée. Anna, assise à leur table habituelle, l'accueille avec un sourire espiègle et lui tend silencieusement le matériel qu'elle a gardé précautionneusement pendant sa détox numérique : sa liseuse qui affiche désormais la couverture du fameux « Marie », et son téléphone portable. Elle pousse aussi devant lui une chemise épaisse simplement étiquetée « Mart-IA-l » sur laquelle trône un mémo : « 11/09 — 14h rdv Quenot ». Souriant, Armand ouvre le dossier et feuillette rapidement les coupures de presse qu'il contient. Il s'arrête sur la photo d'une jeune femme capturée devant un tableau couvert de formules et d'algorithmes, image d'Épinal de la scientifique. Anna explique : « Voici Léonie Quenot, doctorante en informatique, Martial est l'IA née de ses travaux. C'est une IA conçue pour imiter une plume de langue française : elle a été entraînée sur des corpus de la littérature francophone. Elle a aussi été nourrie de biographies d'alchimistes du verbe et de toutes

sortes d'informations sur l'univers du livre. L'éditeur de « Marie » prétend que Martial, l'IA, aurait écrit le roman de son propre chef et l'aurait contacté pour le publier ». « On nage en plein délire » l'interrompt Armand « et que dit Frankenstein ? », ajoute-t-il en pointant du doigt la jeune fille de la photo. « Elle est politiquement plus correcte mais dit peu ou prou la même chose. J'ai pensé que tu aimerais lui parler ».

11 septembre 2026

Armand arrive en avance au rendez-vous. Léonie est déjà là. Après de brèves présentations, la scientifique entraîne l'éditeur dans son bureau, le regarde, le jauge puis se lance : « Votre collègue a évoqué votre parcours atypique, un master en informatique avant de partir dans le monde de l'édition… original… Je me suis dit que vous pourriez comprendre… et m'aider à faire comprendre. » Les heures qui suivent sont pour Armand les plus captivantes qu'il ait passées depuis longtemps. La nuit est tombée depuis longtemps lorsque le silence s'installe. Chacun reste plongé dans ses pensées.

Armand regarde Léonie et devine, derrière la jeune adulte, l'enfant de 10 ans, précoce, avide d'apprendre, dont les prédispositions ont été très tôt remarquées. Encouragée par son enseignante, soutenue par sa famille, Léonie a postulé à un programme spécial proposé par une fondation. Elle a pu bénéficier dès le collège d'une bourse d'études et d'un mentorat sur-mesure. Ainsi la première enveloppe postée un soir de décembre a réussi à ferrer un de ses rêves. Le destin de la seconde a, en revanche, été contrarié. Quelques mois après son envoi, Léonie a reçu une lettre type l'informant du rejet de son manuscrit. Embarquée dans une nouvelle vie, elle a plongé son texte dans une longue hibernation. Voici quelque temps, agacée par les fantasmes autour de l'IA qui agitent la société, une idée iconoclaste a germé en elle : un roman écrit par une IA « autonome » se verrait-il accorder plus de crédit que le même, écrit par une gamine surdouée ? Il ne lui fallut que quelques semaines pour élaborer la mise en scène technique nécessaire. Cette fois, c'est Martial, l'IA et non Léonie, l'enfant qui a soumis son manuscrit au même éditeur. Quel choc ! Le résultat dépassa, et de loin, ses prévisions les plus folles. La créature avait échappé à sa créatrice. L'humain du XXIe siècle, plus performant que le « Turc mécanique » du XVIIIe, avait réussi haut la main une sorte de test de Turing inverse en se faisant passer pour une IA !

Et maintenant ?

Léonie est atterrée d'avoir alimenté les chimères qu'elle voulait combattre.

Armand rompt le silence :

— Révéler la vérité au grand jour va déclencher un cataclysme dans le monde de l'édition, de la presse et bien au-delà : personne n'aime être pris

pour un demeuré. Certains n'y croiront pas, les complotistes vont s'en donner à cœur joie…

— Pourtant il faut bien agir : de telles affabulations ne doivent plus se propager de la sorte… mais que faire ? Je leur ai déjà dit et répété qu'une IA comme Martial n'est pas programmée pour nourrir d'ambition littéraire ! Personne n'a daigné m'écouter ! rétorque Léonie avec frustration.

— Pas étonnant, tente de la rassurer Armand, encore trop peu de personnes ont aujourd'hui les clés scientifiques pour exercer leur esprit critique sur ces questions.

Le silence retombe.

Armand, le regard malicieux, reprend soudain :

— Qui sait… le scandale qui ne manquera pas d'éclater aura peut-être des retombées salutaires… Attendons la publication du palmarès du prix Goncourt. Après tout si Martial est distingué, ce ne sera pas la première mystification réussie. Toi en Romain Gary, Martial en Ém**I**le 5.0 **A**jar, voilà qui pourrait bien bousculer quelques intelligences humaines !

Les pentaminos amis

Jean-Paul Delahaye, professeur émérite
Université de Lille, laboratoire CRISTAL UMR CNRS 9189

La rubrique *Récréations informatiques* propose une petite énigme algorithmique ou sur un thème de mathématiques discrètes susceptible d'intéresser un lecteur de 1024. La solution est donnée dans le numéro suivant.

Rappel du problème précédent — *Magie arithmétique*

La magicienne des nombres n'a qu'un seul rêve : vous étonner. Elle pose sur la table un verre transparent dans lequel un foulard froissé a été enfoncé. Elle vous donne un papier et un crayon et vous invite à choisir un nombre N de quatre chiffres possédant au moins deux chiffre différents. Les nombres 3333 et 5555 ne sont pas autorisés, mais 4359 ou 4455 conviennent. Elle demande :

« Classez les chiffres de N par ordre croissant, cela vous donne un nombre X à quatre chiffres qui commence peut-être par des zéros comme 0035. Classez les chiffres de N par ordre décroissant, cela vous donne un nombre Y à quatre chiffres. Calculez Z = Y - X. Recommencez à partir de Z les mêmes opérations. Faites cela jusqu'à ce que cela devienne inutile car le Z que vous obtenez redonne Z. ».

En quelques secondes vous faites ce qui a été demandé et vous disposez donc d'un résultat Z à quatre chiffres. La magicienne tire alors le foulard du verre et le déploie. Il y est écrit 6174.

C'est effectivement le Z que vous avez trouvé. N'est-il pas paradoxal et magique qu'elle ait pu savoir à l'avance que vous alliez obtenir 6174 ?

À l'aide de votre ordinateur vous trouverez assez facilement la solution, aussi deux autres questions sont posées dont je ne connais pas les réponses :
– peut-on démontrer le résultat sans utiliser d'ordinateur ?
– quelles sont les généralisations envisageables du résultat ?

DOI:10.48556/SIF.1024.26.141

Solution

Merci à Jean-Jacques Pansiot et Éric Wegrzynowski qui m'ont fait parvenir des solutions intéressantes. La réponse est bien sûr, que quel que soit le nombre N de départ, on finit toujours par tomber sur 6174 !

C'est une propriété vérifiable par un calcul fini consistant à essayer tous les N possibles. J'ai écrit un petit programme et je me suis assuré qu'on obtient effectivement toujours Z = 6174. La preuve par l'ordinateur est un peu décevante mais ne laisse aucun doute. Les démonstrations sans utiliser de programme proposées par les lecteurs sont correctes mais un peu longues pour être reproduites ici. Voici quelques informations supplémentaires sur cette bizarrerie des nombres à quatre chiffres qui aujourd'hui ne possède aucune explication générale :

– aucun autre nombre que 6174 ne retombe sur lui-même quand on mène le calcul demandé ;
– le nombre maximum d'étapes de calcul pour arriver à 6174 est 7. C'est par exemple ce qui se passe pour 1400 qui donne successivement : 1400 → 4086 → 8172 → 7443 → 3996 → 6264 → 4176 → 6174 ;
– le nombre moyen d'étapes de calcul avant d'arriver à 6174 est 4,7. Ce nombre d'étapes prend toutes les valeurs possibles entre 0 (pour 6174) et 7 (pour 1400). Le tableau suivant indique précisément combien de nombres X exigent N étapes de calcul :

0	1	2	3	4	5	6	7
1	356	519	2124	1124	1379	1508	1980

Le même miracle se produit encore quand on part d'un nombre de 3 chiffres (non tous égaux). Ils conduisent invariablement à 495. Cette étrange propriété des nombres 6174 et 495 a été découverte par le mathématicien indien Dattatreya Kaprekar (1905-1986) en 1949.

Nous sommes dans une situation où on découvre un résultat inattendu, qu'on prouve facilement par la force du calcul, mais qu'au fond on ne comprend pas. Le mystère est d'autant plus profond que si on essaye la même chose en partant de nombres de 5 chiffres, cette fois ça ne marche plus : tous les nombres conduisent à des cycles — il ne peut pas en être autrement — mais pas nécessairement le même. Éric Wegrzynowski qui a exploré le problème indique : « En opérant avec cinq chiffres, l'itération de la fonction conduit toujours à l'un des trois cycles :

– (63954, 61974, 82962, 75933) ;
– (59994, 53955) ;
– (83952, 74943, 62964, 71973).

De plus, parmi les nombres de cinq chiffres à 1, 2, 3, 4 ou 5 chiffres significatifs ayant au moins deux chiffres distincts, 48480 nombres conduisent au premier de ces cycles, 3190 conduisent au deuxième et 48320 au troisième. »

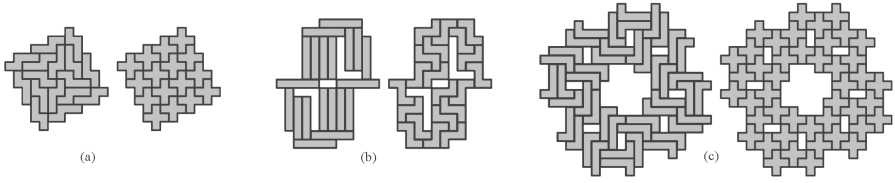
Bibliographie

– Yutaka Nishiyama, Mysterious number 6174[1].
– Brady Haran, Numbers 6174, Numberphile[2].
– Dattatreya Kaprekar, Another solitaire game, *Scripta Mathematica* 15, 244–245, 1949.

Nouveau problème — *Les pentaminos amis*

Les pentaminos sont les formes d'un seul tenant composées de cinq carrés collés par leurs côtés. Il existe 12 pentaminos dont la liste est donnée ci-dessous.

Deux pentaminos sont « amis » si, par définition, il existe une même forme F qu'ils pavent chacun de leur côté.



Le dessin (a) montre que le pentamino N est ami avec le X. Le dessin (b) montre que le I et le Z sont amis, et le dessin (c) montre que le L et le X le sont aussi. Cette notion a été introduite en 1996 par Rodolfo Kurchan dans son livre *Puzzle fun* où il traite quelques cas. Chacune de 66 paires possibles

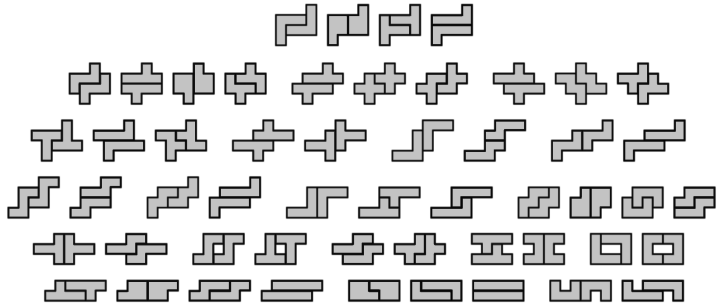
1. https://plus.maths.org/content/mysterious-number-6174, consulté en 2025.
2. https://www.youtube.com/watch?v=d8TRcZklX_Q, consulté en 2025.

de pentaminos crée un problème ludique et parfois difficile. Un bon programme est sans doute nécessaire pour les résoudre.

À vous de les étudier en tentant de remplir le tableau ci-dessous. Dans chaque case du tableau il faut écrire une croix s'il n'y a pas d'amitié possible entre les deux pentaminos (et que vous arrivez à le démontrer), ou un entier donnant le plus petit nombre de pièces prouvant l'amitié. Personne aujourd'hui n'a pu remplir toutes les cases. C'est donc un problème ouvert.

	I	L	N	P	T	U	V	W	X	Y	Z
F											
I											20
L									44		
N									16		
P											
T											
U											
V											
W											
X											
Y											

Les dessins suivants justifient que l'on place quelques « 2 » dans le tableau.



Défi c0d1ngUP
Les cuisines Dentrassis

c0d1ngUP team

Le nouveau défi que nous vous proposons provient de l'édition 2025 de c0d1ngUP et s'intitule *Les cuisines Dentrassis*. Le thème de la 11e édition du challenge de programmation était H2G2, la série de romans de Douglas Adams. Par ailleurs, nous fournissons dans cet article des pistes de résolution pour le défi proposé dans le numéro 25 : *Fabrique de spores*.

Nouveau défi — Les cuisines Dentrassis

> *Les Dentrassis forment une tribu de gourmands indisciplinés, un gros tas de mecs sympa que les Vogons avaient depuis peu choisi d'employer aux cuisines sur leurs flottes au long cours, à la condition expresse qu'ils se tiennent strictement à carreau.*
>
> *Le guide galactique (H2G2 1)*, Douglas Adams[1]

Plutôt que de se bousculer dans les cuisines, les Dentrassis opèrent un roulement, et chaque jour, un seul Dentrassis prépare des plats.

Il utilise un seul four, et tente d'optimiser son temps pour finir au plus vite sa besogne. Pour chaque plat, il connaît à l'avance le temps de préparation, et le temps de cuisson. Chaque plat doit être d'abord préparé puis cuit. Le Dentrassis de corvée ne peut préparer qu'un plat à la fois, mais il peut tout à fait préparer quelque chose pendant qu'un autre plat cuit. Par ailleurs, lorsqu'il commence à préparer un plat, il ne s'interrompt pas (sauf éventuellement pour mettre quelque chose au four) et le termine avant de commencer à en préparer un autre, et lorsqu'il met un plat au four, il ne le ressort pas tant qu'il n'a pas fini de cuire. Enfin, il ne peut cuire qu'un plat à la fois.

1. *Le guide galactique (H2G2 1)*, Douglas Adams, 1979. Traduction de Jean Bonnefoy. Folio SF, ISBN 2-07-041568-6.

DOI:10.48556/SIF.1024.26.145

Supposons qu'il doive préparer les trois plats suivants :
– `M` : milkshake de brume tachyonique et de fruits cosmiques ; prép. 15 min, cuis. 5 min ;
– `I` : infusion de feuilles d'anémone lunaire ; prép. 25 min, cuis. 15 min ,
– `S` : shot de plasma stellaire épicé ; prép. 20 min, cuis 20 min.

Il peut commencer par préparer le milkshake, puis le mettre au four pendant qu'il prépare l'infusion. Au moment de mettre l'infusion au four (oui… c'est une infusion qui va au four, c'est pour les Vogons), le milkshake sera cuit. Il prépare alors le shot de plasma, et enfin le met au four (figure 1).

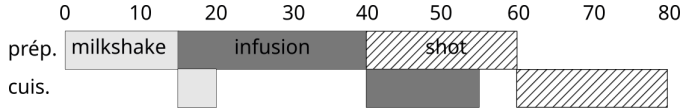


Fig. 1. Premier planning de préparation des 3 plats

En procédant ainsi, au bout de 80 minutes, tout est prêt. Mais il peut s'organiser différemment (figure 2).

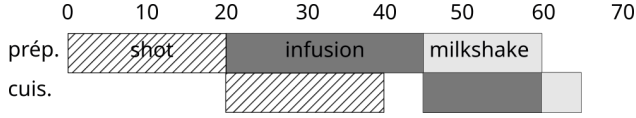


Fig. 2. Deuxième planning de préparation des 3 plats

Et dans ce cas, il sera libre au bout de 65 minutes. Nous donnons aux figures 3 et 4 des plannings qui ne sont pas convenables (ils ne respectent pas les contraintes), si l'on s'en tient aux règles (énoncées plus haut) que s'imposent les Dentrassis.

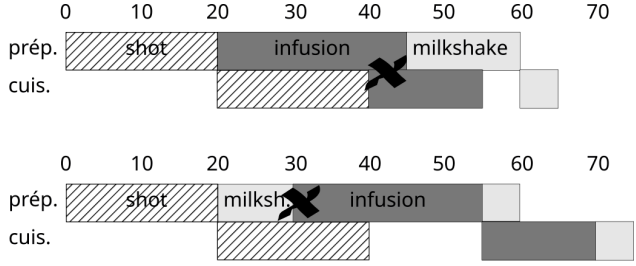


Fig. 3. Deux plannings qui ne respectent pas les règles Dentrassis.

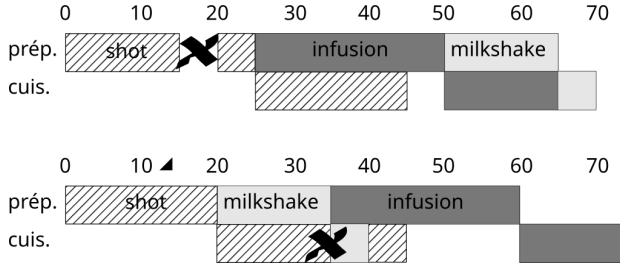


Fig. 4. Deux plannings qui ne les respectent guère plus.

Tous les temps de préparation, ainsi que les temps de cuisson, sont des multiples de 5 minutes, et il est donc possible de décrire un plan de travail en donnant, pour chaque tranche de 5 minutes, deux caractères indiquant quel plat est en cours de préparation et quel plat est en cours de cuisson. Pour la seconde solution, qui ne dure que 65 minutes, on a :

– durant les 5 premières minutes, préparation du shot, et pas de cuisson : `"S."` (en seconde position, un `"."` signifie qu'il n'y a pas de cuisson pendant cette tranche de 5 minutes et, en première position, qu'il n'y a pas de préparation) ;
– idem durant les minutes 5 à 10, 10 à 15 et 15 à 20 : `"S.S.S."` ;
– durant les minutes 20 à 25, l'infusion est en cours de préparation et le shot en cours de cuisson : `"IS"` ;
– …

Toute la séquence est finalement décrite par cette chaîne :

```
S.S.S.S.ISISISISI.MIMIMI.M
```

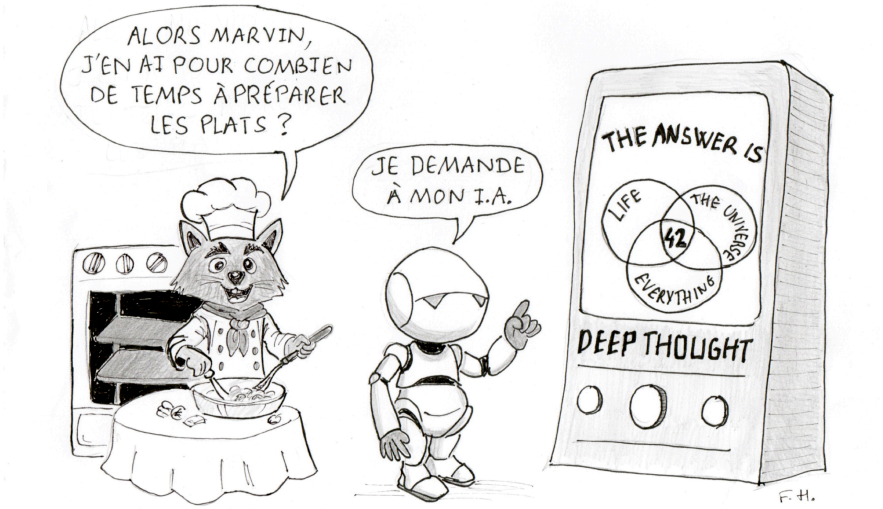
La première séquence, qui dure 80 minutes, serait décrite par :

```
M.M.M.IMI.I.I.I.SISISIS..S.S.S.S
```

Il y a plusieurs séquences différentes qui permettent de terminer le travail en 65 minutes, par exemple : `S.S.S.S.I.ISISISISMIMIMI.M`, mais aucune séquence ne peut libérer le Dentrassis avant 65 minutes.

Pour les remercier de vous avoir pris en stop, vous décidez d'optimiser la prochaine journée pour les Dentrassis. La liste des plats à préparer est disponible en entrée, avec les temps de préparation et de cuisson. Proposez une organisation dans le format utilisé précédemment qui permettra au Dentrassis d'être débarrassé de sa tâche au plus tôt. Attention, chaque couple de lettres du planning vaut pour 5 minutes. Par exemple, le planning `A.BABA.B` signifie :

– minutes 0 à 5 : A en préparation, et four vide ;
– minutes 5 à 10 : B en préparation, A au four ;
– minutes 10 à 15 : B en préparation, A au four ;
– minutes 15 à 20 : aucune préparation, B au four.



Les données d'entrée sont accessibles en téléchargement à cette URL[2] et un extrait est reporté ci-dessous :

```
A : Caviar de poissons de Sandwich IV : prep. 35, cuis. 35
a : Carpaccio de tentacule de poulpe galactique : prep. 5, cui...
B : Beignets de crevettes de la faille de Zorgon : prep. 45, c...
b : Mousseline d'escargots spatiaux de Lamuella : prep. 35, cu...
C : Sushi de kraken cosmique marin : prep. 40, cuis. 30
    ......................
O : Poêlée de bulbes de lunes mordorées : prep. 45, cuis. 40
o : Ratatouille de feuilles d'arbre fractal : prep. 10, cuis. 10
```

Notons que les LLM sont particulièrement doués pour trouver des noms de plats qui auraient pu figurer dans la saga de Douglas Adams…

Vous pouvez nous faire parvenir votre méthode, vos astuces de résolution, vos remarques, accompagnées de la solution au défi, à l'adresse suivante : codingup.team+1024@gmail.com. Dans le prochain bulletin, nous proposerons une synthèse de notre solution et de celles qui nous seront parvenues à temps.

2. `https://codingup.socinfo.fr/1024/26/pb/input_data.txt`.

Solution du défi précédent : Fabrique de spores

Ce problème consistait en un automate cellulaire en 3 dimensions, dont les règles de transition étaient complètement décrites. L'objet du défi consistait à calculer combien de cellules (appelées spores dans le scénario) de chaque type formaient l'automate après la 17e étape de calcul. L'énoncé complet peut être retrouvé dans le numéro précédent de 1024[3].

Commençons par évaluer une solution naïve à ce problème : peut-on représenter l'ensemble des spores jusqu'à la 17e étape. Comme indiqué dans l'énoncé, au départ, on a 8 spores qui forment un cube. À la première itération, 19 nouvelles spores sont déjà créées, donc 27 au total, puis 125 à l'étape suivante, etc. Le nombre de spores après l'étape n est donné par $(2^n + 1)^3$, soit une progression exponentielle[4]. À la 17e étape, on aura près de 2.2×10^{15} spores, qui est une quantité rédhibitoire si l'on envisageait de stocker toutes les configurations, sans parler du temps de calcul nécessaire pour le faire.

Comme alternative, on peut remarquer que l'énoncé ne nous demande « que » de compter le nombre de spores, pas de savoir où elles sont placées. Ainsi, une idée pourrait être, plutôt que de conserver la position de chaque spore, de ne mémoriser que le nombre de cubes de chaque type (il y en a 4^8, soit 65 536 types de cubes). Cette information est suffisante pour déduire le nombre de cubes de chaque type à l'étape suivante, ainsi que le nombre de spores de chaque sorte, qui est l'information qui nous intéresse. Stocker en mémoire une table du nombre de cubes de chaque type à chaque itération devient beaucoup plus commode. Alors, le problème peut se décomposer en trois tâches :

- tout d'abord, créer une table de transition qui, pour chaque type de cube, va donner le type et le nombre de cubes qui vont en découler à l'étape suivante ; c'est cette structure qui va consommer la plus grande quantité de mémoire dans notre algorithme, mais sa taille va rester fixe tout au long des calculs ;
- ensuite, initialiser une nouvelle table contenant le nombre de cubes de chaque type à une itération donnée ; à la première étape, on a un seul cube, celui de départ ;
- finalement, à chaque itération, utiliser la table de transition pour recalculer le nombre de cubes de chaque type.

Évidemment, dans cette représentation, chaque spore va être comptée plusieurs fois. En effet, une spore située au milieu de notre structure sera comptée 8 fois (elle est un sommet de 8 cubes différents). Si elle est située sur un bord, elle ne sera comptée que 4 fois, 2 fois sur une arête, et finalement une seule fois sur un sommet de la structure (voir la figure 5). Ainsi, la position d'une spore est importante après tout.

3. https://doi.org/10.48556/SIF.1024.25.203.

4. OEIS Foundation Inc. (2025), Entry A343318 in The On-Line Encyclopedia of Integer Sequences, https://oeis.org/A343318 : *The number of vertices when starting with a cube (n=0) and iterating by dividing every cube into 8 equal cubes.*

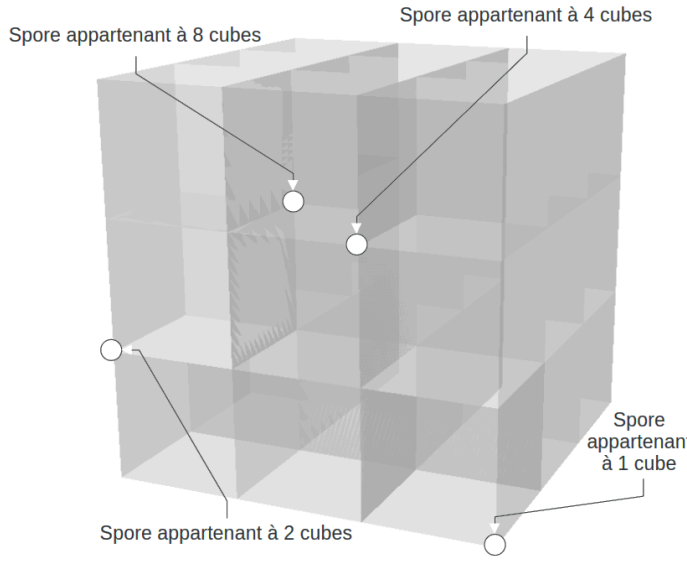


Fig. 5. Nombre de cubes auxquels appartient une spore.

Bien entendu, il est possible de résoudre cette épreuve en maintenant non seulement la table de transition des cubes, mais également des carrés en surface, des segments sur les arêtes, ainsi que des sommets. Mais cela complique le code et multiplie le risque de bugs dans le décompte de tous ces éléments.

Une autre manière de résoudre ce problème est d'ajouter un nouveau type de spores : ajoutées tout autour du cube de départ, elles ont pour propriété essentielle que toute spore en contact avec elles sera dans l'impossibilité de se multiplier. Ainsi, les spores situées sur les bords et les sommets de notre structure sont maintenant bien comptabilisées 8 fois, comme les autres, mais elles ne se multiplieront pas. Cela simplifie grandement le calcul final, au détriment de notre table de transition, car, avec maintenant 5 types de spores au lieu de 4, la table prend de l'embonpoint et comporte 5^8, soit 390 625 entrées.

Et effectivement, une implémentation telle que celle-ci aura la propriété de consommer la plupart des ressources (mémoire et processeur) sur la construction de la table de transition. Le calcul des différentes itérations est comparativement assez peu coûteux. Cela dit, le temps de traitement reste très raisonnable : à titre d'exemple, un code Python implémentant cet algorithme donne une solution en environ 2 secondes sur un ordinateur portable

moderne, 75 % du temps de calcul étant dédiés à la construction de la table de transition initiale.

Cette solution offre tout de même des possibilités d'optimisation. Par exemple, notre table de transition contient tous les cubes possibles, mais sa taille pourrait être réduite afin de tenir compte des symétries et rotations d'un même cube. Toutefois, si cela améliore l'empreinte mémoire, c'est probablement une optimisation dont on peut (et voudra !) se passer dans le contexte d'une compétition en temps limité.

c0d1ngUP team :
Emmanuel Laizé, université de Poitiers (UP),
Freddy Légé (UP), Laurent Signac (UP),
Benoit Tremblais (UP), Loïc Restoux, Caroline Tartary.

La SIF remercie ses soutiens pour leur participation à la diffusion de 1024.

Le Cigref

Le Cigref est une association professionnelle à but non lucratif et sans activité commerciale, dont les 157 membres sont les grandes entreprises françaises et les grandes administrations publiques françaises utilisatrices de produits et services numériques. Les membres du Cigref sont représentatifs de l'économie française et des différents secteurs d'activités qui la structurent (banque, assurance, énergie, distribution, transports, industrie, services, administration…). Les entreprises et administrations publiques membres du Cigref sont représentées par leur dirigeant numérique et technologique (DNum, CIO, CTO, CDO, CITO, CDIO, etc.) La gouvernance du Cigref est assurée par un Conseil d'administration de 15 membres élus par l'Assemblée générale des Représentants des membres du Cigref.

Les 157 membres du Cigref représentent environ :
– 2 000 milliards d'euros de chiffre d'affaires cumulé ;
– 10 millions de salariés servis par les prestations de leurs directions du numérique ;
– 400 000 praticiens du numérique ;
– 70 milliards d'euros de cash out annuel dans l'IT.

Les produits et services numériques développés et déployés par les membres du Cigref comptent, chaque jour, et sur toute la planète, des centaines de millions d'utilisateurs, comme clients, bénéficiaires ou partenaires, à titre personnel ou professionnel.

La raison d'être du Cigref est d'agir au service de ses adhérents et de l'intérêt général afin de bâtir le numérique durable, responsable et de confiance que nous voulons collectivement pour la société et son économie. Sa mission pour y parvenir est de développer la capacité des grandes entreprises et administrations publiques à comprendre, intégrer et maîtriser le numérique. Pour réussir sa mission, l'activité du Cigref s'articule autour de trois axes principaux, qui font sa singularité : l'appartenance, l'intelligence collective et l'influence.

En pratique, le Cigref anime les activités et services suivants au profit de ses adhérents :

– Les groupes de travail, une quinzaine par an, organisés sur les thématiques arrêtées annuellement par le Conseil d'administration. Ils sont animés et synthétisés par un chargé de mission du Cigref, et font l'objet de la rédaction de rapports articulant analyses, retours d'expériences, paroles d'experts et recommandations.
– Les cercles, qui offrent un espace de dialogue et de partage sur des thématiques générales et permettent de transmettre des informations ou des connaissances aux participants, notamment par la mobilisation d'experts. Certains cercles sont explicitement réservés, sur décision du Conseil d'administration, aux seuls Représentants et Représentants délégués.
– Des actions d'influence et de représentant d'intérêts, développées et menées au profit des utilisateurs professionnels de produits et services numériques, auprès des pouvoirs publics, tant au niveau national qu'européen.
– Une démarche de stratégie prospective qui propose un éclairage sur l'avenir du numérique à l'horizon d'une quinzaine d'années et un cadre de réflexion basé sur une méthode de raisonnement rigoureuse.
– Des activités de réflexion, organisées au profit des Représentants et Représentants délégués des membres du Cigref, notamment par la mobilisation de disciplines comme la philosophie, la géopolitique et la prospective.

Inria

Inria est l'institut national de recherche en sciences et technologies du numérique et a la responsabilité depuis janvier 2024 de l'Agence de programmes dans le numérique (algorithmes, logiciels et usages) pour renforcer les dynamiques collectives de la recherche française. La recherche de rang mondial, l'innovation technologique et le risque entrepreneurial constituent son ADN. Au sein de 220 équipes projets, pour la plupart communes avec les grandes universités de recherche, plus de 3800 scientifiques y explorent des voies nouvelles, souvent dans l'interdisciplinarité et en collaboration avec des partenaires industriels pour répondre à des défis ambitieux. Institut technologique, Inria soutient la diversité des voies de l'innovation : de l'édition *open source* de logiciels à la création de startups technologiques (Deeptech). Enfin, Inria promeut depuis plus de 20 ans des actions de médiation scientifique, pour et par le numérique, notamment vers les scolaires et plus particulièrement les jeunes filles, et œuvre fortement pour la science ouverte.

Moteur de la recherche et de l'innovation numérique en France et en Europe depuis plus de 50 ans, Inria veut relever le défi de la compétition en développant notamment une politique de site explicite avec les universités françaises intensives en recherche pour renforcer l'attractivité internationale de ces sites universitaires, leur leadership… et leur impact scientifique, technologique et industriel dans les domaines des algorithmes et de l'informatique quantique, du calcul haute performance, de l'éducation et du numérique, de l'intelligence artificielle, de l'Internet des objets, du logiciel, de la modélisation et de la simulation, du numérique et de l'environnement, de la robotique, de la santé numérique, et des sciences des données.

CNRS Sciences informatiques

CNRS Sciences informatiques contribue à développer, coordonner et faire connaître les recherches dans son périmètre scientifique, qui va de l'algorithmique et du calcul à la robotique et à l'interaction humain-machine, en passant par les sciences du logiciel, l'intelligence artificielle, la science des données et le traitement du signal, des images, des langues et de la parole. Ces recherches se développent en interaction avec les autres sciences (sciences du vivant et du climat, sciences humaines et sociales, ingénierie, technologies quantiques, etc.) et avec des grandes questions de société (sécurité, sobriété, santé, éducation, etc.). Elles donnent souvent naissance à des innovations, au bénéfice du monde socio-économique et culturel.

CNRS Sciences informatiques soutient les travaux de chercheurs et chercheuses engagés dans ces domaines. Avec ses partenaires académiques, il pilote une cinquantaine de laboratoires en France et à l'étranger ainsi que plusieurs programmes de recherche financés par France 2030. Il contribue à la structuration du paysage scientifique national en animant près d'une vingtaine de réseaux thématiques (GDR) et en opérant des infrastructures de recherche, notamment le supercalculateur Jean Zay.

La fondation ANTHONY MAINGUENÉ

Anthony Mainguené, ingénieur informaticien et spécialiste des réseaux et de la cyber-sécurité, était le responsable de la sécurité technique des systèmes d'information au sein de la direction du groupe Bouygues construction à Challenger (Guyancourt). Il est décédé brutalement d'une hémorragie cérébrale à l'âge de 40 ans. Ses compétences et sa conception managériale favorisant le bien-être et l'efficacité au travail, son intégrité et son sens humain, étaient reconnus de tous.

La fondation Anthony Mainguené, à vocation universelle, a été créée afin de transmettre son sens de l'éthique, ses idées et ses espérances. Elle est placée sous l'égide de la Fondation de France et est soutenue par le conseil régional de la Nouvelle Aquitaine.

Le monde change et la place de l'homme avec lui. La fondation souhaite faire prendre conscience que la réflexion éthique est devenue incontournable et ne peut se réaliser pleinement que dans et par l'action. L'éthique permet de s'interroger et d'agir en toute responsabilité, avec engagement, solidarité, équité et respect de l'Autre. Pour ce faire, la fondation intervient selon 5 axes : le droit, le numérique, l'ingénierie, les sciences humaines et sociales, le développement durable.

La fondation Anthony Mainguené promeut les prises de conscience éthiques au moyen de colloques, séminaires, enseignements, remises de prix, et s'attache à réussir le challenge de former des Femmes et des Hommes responsables qui sachent innover et œuvrer en toute liberté de conscience pour relever les grands défis d'une société en mutation.

« *L'humanité est une aventure*
responsable et solidaire qui passe par l'autre. »
Anthony Mainguené

Le syndicat Numeum

Numeum est le syndicat et l'organisation professionnelle de l'écosystème numérique en France. Il représente les entreprises de services du numérique (ESN), les éditeurs de logiciels, les plateformes et les sociétés d'ingénierie et de conseil en technologies (ICT). Numeum rassemble plus de 2 500 entreprises adhérentes qui réalisent 85 % du chiffre d'affaires total du secteur. Présidé par Véronique Torner, Numeum se fixe trois priorités : les territoires, pour accompagner les adhérents en région, les compétences, pour répondre aux défis de la mixité et de l'attractivité, et le numérique responsable pour accompagner et soutenir le développement d'un écosystème numérique dans une trajectoire d'impact positif sur le plan économique, social, sociétal et environnemental à l'échelle européenne, nationale et locale.

Numeum est membre de la fédération Syntec. Le secteur du numérique représente 70 milliards d'euros de chiffre d'affaires et 670 000 employés en France. `http://www.numeum.fr`.

POURQUOI ADHÉRER À LA SIF ?

01.
Rejoindre une communauté ouverte sur le monde

02.
Développer son réseau

03.
Bénéficier d'un accès privilégié aux manifestations proposées par la SIF et ses partenaires

04.
Recevoir le bulletin 1024 dès sa parution

Adhérer c'est aussi

- Contribuer à la légitimité de la SIF en augmentant sa représentativité
- S'engager pour une informatique plus juste, plus éthique, plus responsable

REJOIGNEZ-NOUS !
adhesion.socinfo.fr

FAITES UN DON

Particuliers **Entreprises**

Soutenez nos actions en faisant un don **défiscalisé** à la SIF

don.socinfo.fr

Avec le soutien de

Cigref
RÉUSSIR
LE NUMÉRIQUE

cnrs

Inria

FONDATION ANTHONY MANGUIN
Compétence Conscience Cœur & Corps
Promouvoir des prises de consciences éthiques
Sous l'égide de la Fondation de France

numeum

Sif
Société
informatique de France

Institut H. Poincaré · 11 rue Pierre & Marie Curie · 75231 PARIS Cedex 05