Il faut qu'il y ait en informatique théorique un symbole tel qu'il empêche de calculer

Maël Montévil

Il faut qu'il y ait en informatique théorique un symbole tel qu'il empêche de calculer

Maël Montévil*

14 juillet 2021

Bernard Stiegler se référait souvent à la phrase de Paul Claudel : « Il faut qu'il y ait dans le poème un nombre tel qu'il empêche de compter » (Petit 2019). En informatique, et dans l'idée de donner une place à l'incalculable sans pour autant délaisser le calcul, il a porté l'idée d'introduire dans ce domaine des champs incalculables, notamment pour (re)donner un rôle à la délibération.

L'incalculable est, en un sens, à l'origine de l'informatique notamment avec les théorèmes de Gödel avec lesquels fut introduite la notion de codage. En effet, l'incomplétude démontrée par Gödel signifie que certaines assertions, formulables dans une théorie logique suffisamment riche pour pouvoir traiter l'arithmétique, ne sont ni prouvables ni réfutables dans cette même théorie. L'informatique théorique est donc plus riche et subtile que certaines rhétoriques contemporaines régressives affirmant que tout est calculable. Ces discours portent néanmoins sur une question un peu différente de celle à l'origine de l'informatique. Les théorèmes de Gödel portent sur des aspects purement logico-mathématiques, alors qu'il s'agit plutôt d'interroger le rapport entre l'informatique et les sciences naturelles et sociales. Chez Bernard Stiegler il s'agit plus précisément d'avoir une approche organologique, en abordant l'informatique comme structurante — ou destructrice — pour la noèse, en un mot la pensée en tant qu'elle est aussi la capacité à panser la toxicité d'une situation.

Pour progresser sur la question du rapport entre l'informatique et le calculable, je propose de réinterpréter l'objet de l'informatique théorique puis de faire un détour par la biologie théorique où la question d'un symbole qui empêche de calculer se pose. Enfin, je reviens vers l'informatique en transférant de manière critique certains concepts issus de mes travaux en biologie théorique.

1 L'informatique théorique comme science humaine

L'informatique théorique provient dans une très large mesure des débats en logique mathématique ayant eu lieu au début du XXe siècle. Avec l'apparition de contradictions en mathématiques à la fin du XIXe siècle, des mathématiciens et philosophes se sont tournés vers la logique, et la formalisation de la preuve mathématique, pour asseoir cette dernière sur des bases fiables. Ce projet a cependant été mis à mal par les théorèmes de Gödel montrant les limites intrinsèques des formalismes logiques.

Une des retombées de ces travaux est la conception de l'ordinateur, notamment à travers l'œuvre de Turing. Turing a en effet proposé un formalisme logique basé sur le schéma d'une machine lisant et écrivant sur un ruban. Cette machine formelle est équivalente à d'autres formalismes logiques permettant de définir ce qu'est un calcul, la thèse de Church-Turing posant que les divers formalismes définissant ce qui est calculable conduisent *in fine* à des résultats équivalents. Notons que cette thèse a un statut épistémologique rare dans le domaine des mathématiques. Elle ne peut être prouvée, car il n'y a pas de définition formelle de l'ensemble de ces formalismes. Ce n'est qu'une fois deux formalismes donnés que l'on peut prouver qu'ils sont équivalents et alors l'on dispose d'un théorème, limité à ces deux formalismes. La thèse de Church-Turing concernant l'ensemble des formalismes, elle, n'a pas le statut d'un théorème mais constitue une thèse.

L'ordinateur provient donc d'une problématique logico-mathématique : que peut-on déduire à partir d'axiomes, ou, en termes de machines de Turing, quels processus de calcul se

*https://montevil.org Institut de Recherche et d'Innovation, Centre Pompidou et IHPST, Université Paris 1

terminent ? Nous insistons à ce stade sur un point central : ces cadres mathématiques permettent de comprendre ce que peut la machine. La règle du calcul effectué par la machine, tout comme ses entrées, sont posés par hypothèse. D'un point de vue logique, elle est de l'ordre de l'axiome.

Cette perspective a constitué une fin pour la conception des ordinateurs, machines permettant d'effectuer des calculs au sens de la thèse de Church-Turing. Les calculs effectués par un ordinateur sont alors définis par des programmes, à un haut niveau d'abstraction par rapport à la réalisation matérielle, concrète, de la machine. Une partie importante de l'informatique théorique classique a consisté à élaborer une diversité de formalismes permettant de penser différemment ce qu'est un programme, mais toujours avec l'idée que ces changements formels ne transforment pas ce qui est calculable et ce qui ne l'est pas (à l'exception de langages très simples qui ne permettent pas de calculer toutes les fonctions des machines de Turing).

Dans cette informatique théorique, l'objet théorique est la machine isolée, effectuant des calculs dont les propriétés sont stipulées par ailleurs. Le programmeur est extérieur à la théorie. Si l'on fait une analogie avec les conceptions historiques de la physique où le monde est pensé comme étant écrit par Dieu en langage mathématique, alors le programmeur donne la « loi » de la machine, il est en quelque sorte dans une position divine. Cette perspective signifie aussi que l'on pose que l'on ne dispose d'aucun élément théorique concernant le programmeur.

La distinction entre programmeur et utilisateur est d'ailleurs poreuse. D'un point de vue logico-mathématiques, Turing a introduit le concept de machine de Turing universelle, une machine permettant d'effectuer les mêmes calculs que n'importe quelle autre machine sur n'importe quelle entrée, en codant les propriétés de la machine ciblée dans l'entrée. Ce geste théorique a un sens tout à fait pratique. Qu'il s'agisse du téléchargement d'un programme ou de son entrée par un clavier dans le travail d'un programmeur, une entrée devient une règle de calcul dès lors que le programme est exécuté. En ce sens, il est nécessaire que le programmeur, que l'on associe généralement à la définition de la machine, c'est-à-dire de la règle de calcul, et l'utilisateur, que l'on associe à l'entrée, aient *in fine* le même statut théorique pour ce qui est de l'analyse du calcul. Cette continuité se retrouve par ailleurs dans les approches d'apprentissage machine, en particulier le deep learning. Dans ces approches, le calcul effectué sur une entrée donnée dépend à la fois d'un programme générique mais aussi d'autres données utilisées préalablement en entrée dans la phase « d'apprentissage » de la machine.

Concevoir l'informatique théorique à partir du calcul de la machine isolée pouvait se justifier à l'origine des ordinateurs, lorsque la difficulté principale consistait à faire émerger ce nouveau type de machine. Il se justifiait d'autant plus que l'ordinateur se posait comme la machinisation, l'exosomatisation chez Stiegler, de la partie de l'esprit humain que des philosophes comme Frege pensaientt comme étant la plus rationnelle et la plus sûre, spécifiquement la logique. Aujourd'hui, où les ordinateurs sont omniprésents, sous des formes diverses, et mis en réseaux, cette perspective nous semble par contre bien insuffisante, car elle ne prend pas en compte les conséquences des ordinateurs sur la noèse.

Ma réponse à l'appel de Bernard Stiegler à refonder l'informatique théorique consiste alors à changer son objet. Plutôt que de considérer la machine déroulant son calcul de manière isolée, il s'agit de considérer les machines en lien avec les êtres noétiques.

Ce geste théorique a plusieurs conséquences immédiates. La première est que l'informatique théorique ne doit pas se limiter à considérer les effets des programmeurs et utilisateurs sur les machines mais doit aussi considérer les effets des machines sur ces derniers. En particulier, étant donné que les ordinateurs dépendent des savoirs humains pour exister, si l'informatique conduit à une prolétarisation excessive, c'est-à-dire à une perte de ces savoirs, alors elle risque de conduire à la destruction de ses propres conditions de possibilité. L'informatique théorique pourrait bien alors posséder une cohérence interne, mais elle serait pourtant fondamentalement irrationnelle.

La seconde conséquence de ce geste est que l'informatique théorique classique ne traite alors que d'un cas très particulier, le cas où la machine est laissée seule à son calcul. Or ce n'est pas ce que font les machines concrètes, elles sont utilisées de manière interactive et leurs programmes sont couramment transformés. Il s'ensuit que l'informatique théorique classique ne permet pas de comprendre les trajectoires réelles (observables) de ces objets que sont les ordinateurs. De ce point de vue, l'informatique théorique classique deviendrait un cas limite d'une nouvelle informatique théorique, de même que la mécanique classique est un cas limite de la relativité générale, le cas où les vitesses et les masses sont faibles. Dans

le cas de l'informatique, il s'agirait du cas où l'input et la programmation de la machine est donnée, et la machine calcule de manière isolée.

Il existe bien des approches théoriques pour traiter les situations de parallélisme, par exemple dans la situation où plusieurs utilisateurs en ligne essayent de commander une seule place de concert disponible, mais ces approches se limitent à faire en sorte que dans tous les cas le programme se déroule de manière conforme aux fins du programmeur (et de son employeur ou commanditaire). En l'espèce il s'agit de faire en sorte qu'un seul utilisateur puisse payer pour cette unique place disponible – le problème est alors équivalent aux enjeux techniques du parallélisme (plusieurs calculs effectués en parallèle, de manière asynchrone, ce qui introduit de l'aléatoire tout comme les activités des utilisateurs agissant en parallèle), qui sortent du cadre strict des machines de Turing, lesquelles sont déterministes. Ces approches sont donc très loin de théoriser l'activité des utilisateurs.

Par contre, il existe un cadre, ou plutôt la convergence de deux cadres, que l'on peut considérer comme une esquisse d'informatique théorique étendue – une esquisse biaisée et particulièrement pathogène. Il s'agit de la convergence entre l'informatique et les sciences cognitives, telle qu'enseignée à Stanford, qui est à la base de nombreuses plateformes et mécaniques de jeux vidéo conçues pour rendre l'utilisateur addict, et plus généralement pour que son comportement suive les intérêts de l'éditeur. Cette convergence ne pense pas la question du développement biologique et psychologique, bien qu'elle vise dans certain cas l'éducation, et elle n'aborde pas la question de la noèse, la pensée, au-delà de quelques propriétés très simples. Elle signale cependant l'importance d'envisager une alternative théorique à cette convergence, alternative qui ne s'oriente pas vers une opportunité économique à court terme mais vers un soin apporté à l'informatique et à la noèse.

Pour repenser l'informatique théorique, faisons un détour par la biologique théorique. Dans ce domaine, certains concepts, idées et questions peuvent permettre d'enrichir la réflexion sur l'informatique, et ceci notamment dans sa relation à l'écriture mathématique.

2 Il faut qu'il y ait en biologie théorique un symbole tel qu'il empêche de calculer

Dans cette partie, nous allons aborder certaines questions de biologie théorique pour arriver à l'introduction récente d'un symbole ayant une épistémologie qui nous semble novatrice (Montévil et Mossio 2020).

Le point de départ de cette analyse est la théorisation de l'historicité du vivant, bien évidemment dans la lignée de la théorie de l'évolution. Contrairement à la génétique des populations, ce qui nous intéresse ici n'est pas la mathématisation de certains mécanismes évolutifs. Il s'agit plutôt de nous concentrer sur la contrepartie théorique et épistémologique de ce caractère historique du vivant, notamment concernant sa mathématisation.

La mathématisation en sciences de la nature provient historiquement de la physique et c'est dans ce domaine qu'elle est la plus développée. Pour décrire rapidement la situation, cette mathématisation passe par l'idée que le changement ayant lieu dans un phénomène puisse être compris sur la base d'une invariance plus fondamentale que ce changement (Bailly et Longo 2006). L'espace des possibles est donné à l'avance et la trajectoire ou la structure finale d'un phénomène dérive de structures prédéfinies, telles que les symétries données par des principes théoriques — par exemple, les symétries de l'espace-temps des relativités Galiléenne, restreinte ou générale.

Prendre au sérieux l'historicité du vivant signifie, à mon sens, que l'on ne peut plus comprendre le changement par l'invariance. Au contraire, il faut comprendre comment, non seulement la forme des êtres vivants change, mais aussi leurs physiologies, leurs modes de reproductions, leurs fonctions et *in fine* les invariants que l'on semble parfois pouvoir distinguer en regardant certains spécimens. Il s'agit alors de postuler l'historicité, et notamment le fait que les êtres vivants puissent varier en un sens fort, c'est-à-dire sans invariance sous-jacente (Montévil, Mossio et al. 2016). Ceci étant posé, il ne s'agit pas pour autant d'abandonner, pour la biologie, le concept d'invariance, mais celui-ci n'est plus premier. Une invariance donnée est alors locale, limitée à une famille plus ou moins large d'êtres vivants, et contingente au sens où certains d'entre eux peuvent la faire varier. Nous avons appelé ces invariants locaux des contraintes (Montévil et Mossio 2015 ; Soto et al. 2016).

Les contraintes ont plusieurs rôles théoriques. Elles canalisent et structurent des processus de transformations. Par exemple, l'ADN comme contrainte canalise les processus de production de protéines, ou les os du bras limitent les mouvements possibles pour ce dernier.

Ce faisant les contraintes rendent possibles des processus qui n'auraient pas lieu sans ces contraintes. Ainsi, sans l'ADN, des protéines formées aléatoirement ne seraient que rarement fonctionnelles, et sans les os du bras, la plupart de ses mouvements ne seraient pas possibles. Les contraintes limitent aussi, notamment, l'état par défaut des cellules : la prolifération et la motilité.

Les contraintes d'un organisme ont une autre propriété remarquable, elles se maintiennent collectivement, par le biais des processus qu'elles contraignent. Ainsi, les séquences d'ADN contraignent la transcription de l'ARN messager, lequel contraint la production de protéines, et, parmi ces dernières, certaines contraignent divers processus, maintenant la structure de l'ADN. Le même type de circularité se retrouve à des niveaux beaucoup plus macroscopiques, par exemple entre les organes d'un vertébré. Cette propriété, que l'on a appelé clôture entre contraintes, ne signifie pas que l'organisme se maintienne à l'identique. Il doit juste maintenir ses contraintes pour que ces dernières durent face à la croissance spontanée de leur entropie, et donc à la disparition de leur invariance (qui reste locale et en un sens contingente).

Enfin, les contraintes jouent un rôle causal diachronique au sens où elles permettent l'apparition de nouveautés. Par exemple, les mâchoires articulées ont permis l'apparition de dents, et toutes sortes de fonctions comme la protection des œufs chez certains poissons à nageoire rayonnée (par exemple *Opistognathus aurifrons*), le transport de petits, ou la parole articulée chez *Homo sapiens*.

Si les contraintes permettent d'aborder certains aspects d'un organisme donné, dans un contexte donné, elles ne permettent pas de définir cet organisme. Rappelons qu'en physique la définition théorique d'un objet est donnée par ses invariants et symétrie, et plus généralement par un cadre théorique mathématisé. Il s'ensuit que l'objet théorique est générique au sens où tous les électrons, par exemple, suivent les mêmes équations – ils n'ont aucune singularité au sens philosophique du terme. Ceci a des conséquences très pratiques. La vitesse de la lumière est définie comme la vitesse de n'importe quel rayon lumineux dans le vide (ou de n'importe quel photon d'un point de vue corpusculaire). La capacité à définir ainsi les objets sur le plan de la théorie permet aussi une certaine séparation entre l'objet concret et l'objet théorique. Il n'est pas nécessaire d'ancrer l'objet théorique à un objet concret particulier (le mètre étalon n'a une utilité que pratique, s'il était détruit il pourrait être reconstruit). En biologie, nous ne possédons pas de telles constructions théoriques parce que, d'un part, les organismes sont constitués d'une multiplicité de contraintes particulières qui sont apparues au cours du temps et d'autre part ces contraintes continuent à changer, même dans des conditions standardisées de laboratoire (Montévil 2019a).

Il est alors intéressant de rappeler l'originalité épistémologique remarquable de la méthode phylogénétique de classification du vivant (Lecointre et Le Guyader 2006). Dans cette méthode, les biologistes n'ambitionnent pas de décrire les êtres vivants par les relations entre leurs parties et l'invariance de ces relations, comme pour les systèmes physiques. Ils définissent les groupes comme étant l'ensemble des descendants d'un ancêtre commun. Ce dernier est théorique, il n'est pas identifié concrètement. En pratique, les spécimens sont donc mis en relation par leurs apparentements estimés. Ainsi, les mammifères possèdent des caractères communs que les oiseaux n'ont pas, ils ont donc très vraisemblablement un ancêtre commun que n'ont pas les oiseaux et forment un groupe. Puisque les objets ne peuvent pas être décrits par des invariants provenant de leur détermination théorique, les biologistes s'appuient sur un autre type d'invariance, celle du passé commun à ces objets, passé défini par la généalogie sous-jacente à la théorie de l'évolution. En définissant les objets par leurs passés, cette perspective permet de ne limiter en rien les variations que le cadre théorique permet d'accueillir.

Cette méthode comporte un second point qui pour nous est significatif. La définition opératoire d'un groupe ne peut pas passer par un ancêtre commun car celui-ci est inconnu, ni par l'invariance de sa structure causale, car cette dernière varie. Elle passe donc par la référence à un spécimen particulier, appelé un holotype. Ce dernier n'est pas l'ancêtre commun, qui sert à définir un groupe comme étant l'ensemble descendance, il est un point de référence permettant de fixer le sens d'un nom dans la classification. Contrairement à la physique, la désignation d'objets concrets est donc nécessaire dans l'épistémologie de cette classification, et, par extension, à l'épistémologie de la biologie, car la classification donne les noms des objets qu'elle étudie. Bien évidemment, dans la pratique expérimentale en biologie, d'autres éléments peuvent être ajoutés à la définition d'un objet biologique, tel que la généalogie connue lorsqu'elle correspond à des animaux élevés en laboratoires, et le milieu dans lequel ils vivent. Ces aspects ne changent cependant pas l'enjeu épistémologique qui consiste à définir dans une large mesure les objets par leur passé plutôt que par ce qu'ils

font (Montévil 2019a).

L'écriture mathématique, en physique, est fondée sur l'invariance des relations entre grandeurs pertinentes. Ceci ne correspond pas aux conditions théoriques et épistémologiques de la biologie. Pour s'appuyer sur l'épistémologie de l'historicité esquissée ci-dessus, nous avons introduit un nouveau type de symbole en biologie théorique, noté χ (Montévil et Mossio 2020). Il s'agit ici d'un symbole plutôt que d'une grandeur, ou d'une variable, car l'épistémologie de ce symbole passe par la référence à un objet concret, par exemple un type de la classification phylogénétique (mais cette approche générale peut être adaptée à la diversité des situations rencontrées pour en tenir compte avec précision).

Ce symbole remplit plusieurs rôles épistémologiques. Il permet tout d'abord de rendre compte explicitement, au sein de la description formelle des objets, de la définition d'un objet par son passé. Mais il vise aussi à accueillir dans ces formalismes l'éventualité de variations dont la nature ne peut pas être prédite, l'apparition de nouveauté en un sens fort (Montévil 2019b). Il ne s'agit pas pour autant d'abandonner les contraintes, comme invariance locale, mais de les articuler formellement à ce type de symbole. Ceci permet, par exemple, de rendre compte des contraintes dont la fonction principale est d'engendrer des nouveautés, sans pour autant que la nature de ces dernières soit donnée par avance. L'articulation de χ et des contraintes permet d'historiciser ces dernières explicitement, par contre ce cadre implique que la validité d'une contrainte précise peut toujours être remise en question par une variation possible, ces dernières étant bien évidemment plus ou moins fréquentes et importantes suivant les contraintes considérées.

Le symbole χ rend donc compte d'une partie de la détermination théorique de l'objet biologique qui n'est pas saisie par une invariance sous-jacente et qui de ce fait ne permet pas le calcul, notamment pour ce qui est de prédire l'apparition de nouveautés fonctionnelles. L'originalité de cette approche est l'articulation de cet incalculable à des considérations épistémologique et méthodologiques précises rendant compte d'éléments essentiels de la biologie : la classification du vivant mais aussi certains aspects des pratiques expérimentales. Ces considérations sont par ailleurs fréquemment oubliées dans d'autres domaines de la biologie, la biologie expérimentale typiquement, à cause de perspectives épistémologiques héritées de la physique sans que ces domaines ne réunissent toutefois les conditions théoriques de ces théories physiques. Ici, l'utilisation d'un nouveau symbole provient donc du croisement de deux épistémologies, l'épistémologie relationnelle de la modélisation mathématiques telle que pratiquée en physique, et se manifestant ici à travers le concept de contraintes, et l'épistémologie historique issue notamment de la biologie de l'évolution.

3 Vers une nouvelle informatique théorique

Nous souhaitons maintenant suggérer que certains concepts de biologie théorique pourraient être mobilisés pour donner de nouvelles perspectives à l'informatique théorique. Ce type de discussion demande cependant toujours un recul critique. Certains concepts, comme l'état par défaut des cellules n'ont pas de contrepartie évidente en informatique. Par contre, d'autres concepts participent d'une réflexion sur l'articulation entre historicité et mathématiques, et leur pertinence est plus immédiate. En un sens, l'historicité est propre au vivant, mais l'étude du vivant ne se limite pas à la biologie. Les êtres humains, les sociétés humaines, et les artefacts qu'ils produisent participent aussi du vivant, avec des particularités théoriques, notamment la noèse. Bien évidemment nous nous somme concentré, dans la discussion précédente, sur des questions de biologie théorique dont nous pensons qu'elles ont une pertinence pour repenser l'informatique.

Le premier concept que nous pensons pertinent est le concept de contrainte. Une contrainte est d'abord une invariance locale, maintenue loin du maximum d'entropie. Ainsi, le hardware d'un ordinateur ou d'un smartphone est une contrainte à la fois sur la manière dont l'énergie libre sous forme électrique, venant du secteur ou d'une batterie, est dissipée. Ils constituent aussi une contrainte pour les programmeurs et utilisateurs car ils ne changent pas et ont néanmoins un rôle de nature causale vis-à-vis des processus ayant lieu. Ajoutons que, toujours au niveau du matériel, le concept de clôture entre contraintes a une pertinence. Le matériel doit être maintenu, que cela soit par l'entretien, souvent limité à retirer la poussière s'accumulant dans la ventilation d'un ordinateur, ou plus souvent par le remplacement de composants ou des machines entières — ces dernières subissent la croissance de leur entropie, ce qui conduit à leurs dysfonctionnements. Les composants où ces phénomènes sont les plus perceptibles sont les batteries, dont la capacité diminue au cours du temps, et les

périphériques de stockage, comme les disques durs et les SSD qui fonctionnent grâce à une certaine redondance tel que l'utilisation de secteurs supplémentaire permettant de remplacer les secteurs devenus défectueux. Le matériel a aussi un rôle diachronique au sens où il contribue à rendre possible l'apparition de nouvelles contraintes, y compris la production de nouveau matériel (aujourd'hui il faut des ordinateurs pour construire des ordinateurs).

Le concept de contrainte est aussi pertinent pour comprendre le logiciel. Le logiciel contraint, notamment, l'activité de l'utilisateur, mais il ne le détermine pas aussi fortement que les principes physiques déterminent le comportement d'un objet en physique. Comme dans le cas du matériel, le code d'un logiciel est maintenu activement, notamment par les processus de copie leur permettant de durer au-delà de la durée de vie de leur support. Ce que les programmeurs appellent le maintien d'un logiciel est cependant distinct, il s'agit de s'assurer qu'ils fonctionnent toujours alors que certains des logiciels dont il dépend changent, et aussi de corriger les failles de sécurité qui peuvent être détectée. Penser le logiciel comme contrainte signifie que, tout comme la géométrie et la rigidité des os d'un bras à la fois contraignent et rendent possible son mouvement, le logiciel à la fois contraint ce qui est possible tout en permettant ou en facilitant certains processus. En biologie, certaines contraintes ont d'abord comme fonction de maintenir d'autres contraintes tandis que d'autres, appelées contraintes propulsives, ont un rôle de nature plus fondamentalement diachronique, participant de l'apparition de nouvelles contraintes (Miquel et Hwang 2016). Notons que, transposées dans ce vocabulaire, les rétentions tertiaires, comme l'écriture, sont elles-mêmes aussi des contraintes.

À ce stade, il est intéressant de comparer les concepts de contrainte et de pharmakon. Ces concepts ne recouvrent pas tout à fait les mêmes questions, le concept de contrainte étant plus local — il ne comprend pas par lui-même la question du rôle de ces contraintes dans une organisation. Néanmoins, les contraintes ont l'ambivalence du pharmakon au sens où une contrainte limite les possibles tout en les constituant. En l'espèce, la question de l'ouverture ou de la fermeture des possibles concernant les logiciels est une question éminemment pharmacologique ... et pressante. Si l'articulation entre informatique et sciences cognitives mentionnée plus haut sert d'abord à accorder le comportement de l'utilisateur aux intérêts de l'éditeur, c'est typiquement par des stratégies basées sur une addiction pathologique, ou la capacité des utilisateurs à produire des nouvelles possibilités est fortement dégradée. Ces questions renvoient naturellement à celle du design, et à celle des fins du design.

Notons que, du côté de la programmation, l'informatique théorique classique ne tient un propos précis que concernant le fonctionnement des programmes, laissant donc ainsi de côté les changements de ces programmes, c'est-à-dire les processus de programmation. En un sens, ces changements constituent pourtant paradoxalement une de leurs préoccupations centrales. Nous avons vu, avec la thèse de Church-Turing, que tous les formalismes sont considérés comme équivalents pour ce qui est de ce qu'ils permettent de calculer. Dans la pratique concrète, cela signifie que tous les langages suffisamment riches permettent de calculer les mêmes fonctions. Pourquoi, alors, introduire de nouveaux formalismes et langages ? La raison principale, à notre sens, est que ces différentes approches du calcul permettent de traiter les problèmes suivant des perspectives différentes, et que certains problèmes sont plus aisés à aborder suivant une perspective ou une autre. Dans la pratique, les langages peuvent, de plus, avoir un niveau d'abstraction plus ou moins élevé par rapport à ce qu'il se passe dans l'architecture matérielle concrète, l'abstraction ayant des avantages comme la facilité et la portabilité vers différentes architectures, et des inconvénients comme un contrôle moins précis des processus et généralement une rapidité moins grande. Penser les langages de programmation eux-mêmes, et plus spécifiquement leur implémentation dans un logiciel interprétant le code tel qu'un compilateur, comme des contraintes permet de surmonter ce paradoxe. Ils agissent comme contrainte à la fois pour la compilation ou l'exécution du code et sur l'activité des programmeurs.

Cette analyse est aussi pertinente pour ce qui est du code lui-même, lequel agit comme une contrainte sur deux processus distincts. Le code définit un logiciel comme contrainte, et en même temps, il joue le rôle d'un texte lisible par les pairs. Ce deuxième rôle se manifeste notamment à travers les commentaires qui n'ont pas de rôle pour l'exécution du code mais servent à en faciliter la compréhension. Si cette compréhension vise parfois un rôle pédagogique, elle vise aussi et surtout à permettre de retravailler ce code, donc à le changer. Les commentaires jouent donc un rôle diachronique, autrement dit, ils constituent des contraintes propulsives. De même, ce double rôle apparaît pour la partie du code utilisée par la machine à travers un compromis fréquent entre l'optimisation du calcul et sa lisibilité.

Penser l'informatique à travers le concept de contrainte vise aussi à repenser le lien entre

informatique et mathématiques. L'informatique théorique classique émane des mathématiques, et les mathématiques utilisées sont pour l'essentiel discrètes. Elles correspondent à des situations où la mesure peut être en principe parfaite et la détermination est laplacienne comme le souligne Turing lui-même (Turing 1950). À l'opposé, nous avons introduit le concept de contrainte précisément pour rendre compte des limites de la description mathématique des objets biologiques, limites dues au croisement de deux épistémologies distinctes, celle de l'historicité d'une part et celle des relations entre les parties d'un système de l'autre. Introduire ce concept en informatique signifie alors que l'objet théorique de l'informatique ne suit pas un cadre mathématique stable, mais comporte une historicité fondamentale. Si l'on considère un ordinateur donné, la trajectoire suivie n'est plus alors le déroulement d'un programme sur une entrée donnée, mais une relation permanente entre des contraintes exosomatiques (le matériel, les logiciels) et l'utilisateur. A fortiori, ce point de vue est essentiel lorsque l'utilisateur change le code des logiciels qu'il utilise — ou, de manière plus rare mais néanmoins essentielle, lorsqu'il participe à concevoir et construire du matériel.

Ceci nous conduit alors, à envisager d'introduire en informatique théorique quelque chose comme le symbole χ introduit en biologie. Nous ne disposons pas encore d'un cadre élaboré pour ce faire, mais nous pouvons introduire certaines remarques. Ici, la contribution de la classification phylogénétique du vivant n'est plus réellement pertinente, mais la définition de l'utilisateur par son histoire peut l'être — rejoignant ainsi la médecine où l'histoire du patient est essentielle. La manipulation théorique de χ dépend des enjeux à traiter. Par exemple, χ permet de porter l'idée que les savoirs ne sont jamais d'ordre purement synchronique, ils sont d'abord diachroniques. En tant que tels ils sont surtout portés par des personnes et des groupes précis, ce qui rejoint l'utilisation des types dans la classification en biologie.

Pour conclure, l'informatique théorique peut être vue sous deux angles qui, bien que distincts, sont fortement liés. Elle peut être un cadre pour concevoir les machines et les logiciels et elle peut être aussi un cadre pour comprendre ce que font ces machines. On pourrait objecter à l'idée de penser l'informatique avec le concept de contrainte, que ce concept est surtout pertinent pour ce deuxième sens d'informatique théorique, orienté vers la compréhension. Or ce n'est précisément pas l'enjeu ici, car une théorie permettant une compréhension plus précise de ce que font les ordinateurs vise bien à alimenter la pratique en laissant de côté une conception réductionniste de l'informatique où seul compterait, *in fine*, la machine isolée et ses capacités alors que l'utilisateur et le programmeur sont considérés comme radicalement inconnus. Contre cette dichotomie, refonder l'informatique théorique vise donc à réinsérer la noèse dans l'informatique comme question fondamentale pour le travail des informaticiens.

Références

Bailly, F. et G. Longo (2006). *Mathématiques & sciences de la nature : la singularité physique du vivant*. Paris : Hermann.

Lecointre, G. et H. Le Guyader (2006). *Classification phylogénétique du vivant*. Belin.

Miquel, P.-A. et S.-Y. Hwang (2016). « From physical to biological individuation ». In : *Progress in Biophysics and Molecular Biology* 122.1, p. 51-57. ISSN : 0079-6107. DOI : 10.1016/j.pbiomolbio.2016.07.002.

Montévil, M. (avr. 2019a). « Measurement in biology is methodized by theory ». Anglais. In : *Biology & Philosophy* 34.3, p. 35. ISSN : 1572-8404. DOI : 10.1007/s10539-019-9687-x. URL : https://montevil.org/publications/articles/2019-Montevil-Measurement-Biology-Theory/.

Montévil, M. (nov. 2019b). « Possibility spaces and the notion of novelty : from music to biology ». Anglais. In : *Synthese* 196.11, p. 4555-4581. ISSN : 1573-0964. DOI : 10.1007/s11229-017-1668-5. URL : https://montevil.org/publications/articles/2019-Montevil-Possibility-Spaces-Novelty/.

Montévil, M. et M. Mossio (mai 2015). « Biological organisation as closure of constraints ». Anglais. In : *Journal of Theoretical Biology* 372, p. 179-191. ISSN : 0022-5193. DOI : 10.1016/j.jtbi.2015.02.029. URL : https://montevil.org/publications/articles/2015-MM-Organisation-Closure-Constraints/.

Montévil, M. et M. Mossio (juin 2020). « The Identity of Organisms in Scientific Practice : Integrating Historical and Relational Conceptions ». Anglais. In : *Frontiers in Physiology* 11, p. 611. ISSN : 1664-042X. DOI : 10.3389/fphys.2020.00611. URL : https://montevil.org/publications/articles/2020-MM-Identity-Organism/.

Montévil, M., M. Mossio et al. (août 2016). « Theoretical principles for biology : Variation ». Anglais. In : *Progress in Biophysics and Molecular Biology* 122.1, p. 36-50. ISSN : 0079-6107. DOI : 10.1016/j.pbiomolbio.2016.08.005. URL : https://montevil.org/publications/articles/2016-MMP-Theoretical-Principles-Variation/.

Petit, C. (mai 2019). « L'impact de la technologie sur l'emploi, l'économie et la société (4). Entretien avec Bernard Stiegler ». In : *Revue du MAUSS permanente,* URL : http://www.journaldumauss.net/./?L-impact-de-la-technologie-sur-l-emploi-l-economie-et-la-societe-4-Entretien.

Soto, A. M. et al. (août 2016). « Toward a theory of organisms : Three founding principles in search of a useful integration ». In : *Progress in Biophysics and Molecular Biology* 122.1, p. 77-82. ISSN : 0079-6107. DOI : 10.1016/j.pbiomolbio.2016.07.006. URL : https://montevil.org/publications/articles/2016-SLN-Conclusion-Century-Organism/.

Turing, A. M. (1950). « Computing machinery and intelligence ». In : *Mind* 59.236, p. 433-460.