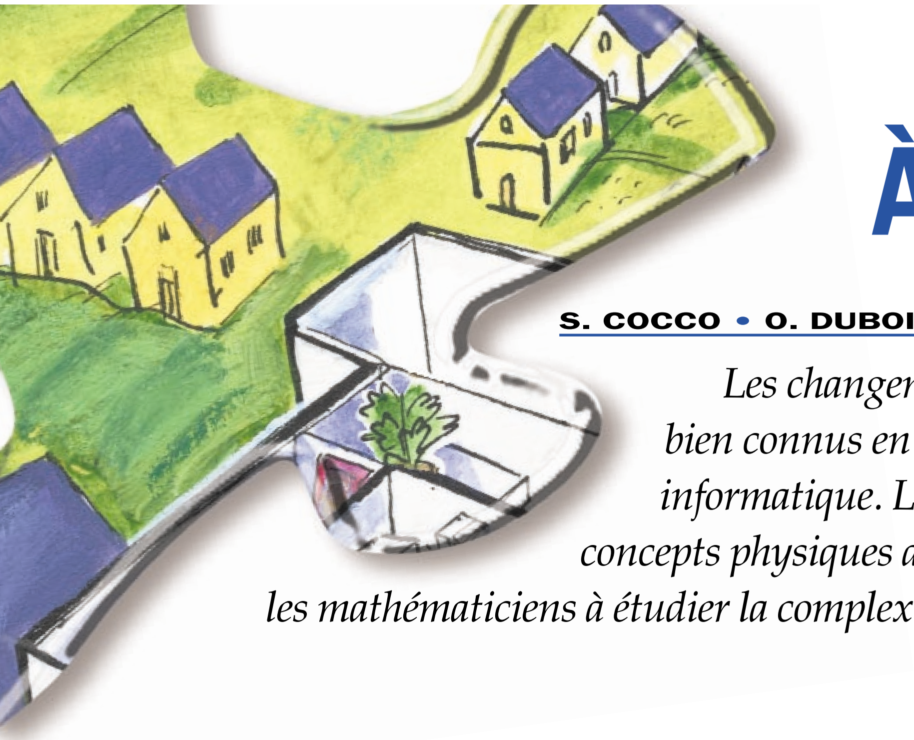


À la rescousse

S. COCCO • O. DUBOIS • J. MANDLER • R. MONASSON

Les changements brusques de comportement, bien connus en physique, se rencontrent aussi en informatique. La transposition de l'analyse et des concepts physiques aux problèmes d'optimisation aide les mathématiciens à étudier la complexité de résolution de ces problèmes.

Un particulier consulte un architecte pour la construction d'une maison. Plusieurs options lui sont offertes : une maison de plain-pied ou à un étage, un patio, une terrasse et un garage. Ses goûts et son banquier lui dictent plusieurs contraintes : il faut qu'il y ait un étage ou une terrasse, mais pas les deux ; il faut qu'il y ait un garage ou un patio, mais pas les deux ; s'il y a un patio, la maison doit être de plain-pied ou avoir une terrasse (ou les deux) ; si la maison est de plain-pied ou a une terrasse, il ne faut pas qu'il y ait de garage. Une maison peut-elle être construite?

Cette énigme logique illustre un problème fondamental en informatique, le problème de satisfaisabilité, ou problème SAT, qui est un problème de nature combinatoire, et dont les applications sont nombreuses, par exemple, pour la conception assistée par ordinateur de microprocesseurs, pour la vérification des circuits électroniques, en intelligence artificielle et en robotique.

Ces applications du problème SAT, dont l'expression formalisée, nommée formule SAT, est parfois de taille considérable, ont motivé la recherche d'algorithmes efficaces pour les résoudre. Par efficacité, on entend la complexité de résolution, soit le temps maximal nécessaire à la résolution par un algorithme de tout problème du même type. La complexité de résolution des formules SAT apparaît souvent exponentielle, c'est-à-dire que le temps de résolution croît exponentiellement avec le nombre de données. Cependant, certaines formules SAT sont parfois résolues par des algorithmes de complexité polynomiale : le temps de résolution croît selon une puissance fixée du nombre de données. Dans le premier cas, les limites des ordinateurs, même les plus puissants, sont vite atteintes. Dans le second, la résolution de formules SAT de très grande taille reste accessible. Les mathématiciens s'attachent à caractériser les formules les plus difficiles à résoudre et à prévoir le temps nécessaire pour les résoudre.

Ces deux tâches sont du ressort de la théorie de la complexité algorithmique, une discipline née à la fin des années 1970 après que la croyance en la toute puissance des ordinateurs s'était heurtée au mur des réalités. Il s'agit de distinguer les problèmes traitables, dont l'algorithme de résolution est de complexité polynomiale, de ceux intraitables, pour lesquels on dispose seulement d'algorithmes de complexité exponentielle. Un autre exemple classique de problème intraitable est le problème du voyageur de commerce qui, chargé de visiter plusieurs villes une seule fois chacune doit minimiser son parcours *(voir l'encadré de la page xx).*

Beaucoup de problèmes intraitables ont une importance pratique, aussi, améliorer leur connaissance est devenu l'un des thèmes majeurs de l'informatique théorique ces 20 dernières années. Ils donnent lieu à des études tant théoriques qu'expérimentales où le problème SAT joue un rôle privilégié, en raison notamment de sa simplicité d'énoncé et de sa situation au carrefour de plusieurs disciplines, telle la logique et les mathématiques.

Récemment, des expérimentations sur la résolution de formules SAT, ainsi que sur d'autres problèmes de combinatoire, ont révélé un étrange phénomène de seuil *(voir la figure 3)*, qui s'apparente aux transitions de phases, telle la fusion de la glace ou la vaporisation de l'eau. Cette analogie a ouvert de nouvelles pistes pour l'étude de la difficulté de résolution des problèmes SAT. Les mathématiciens se sont alliés avec succès aux physiciens pour aborder ce problème : les premiers ont mis en évidence l'existence d'un seuil et son lien avec la difficulté de résolution d'un problème SAT, les seconds en ont révélé la signification. Cette association féconde s'est concrétisée par l'amélioration des algorithmes de résolution.

Le problème SAT

Exprimons le problème SAT de notre acheteur immobilier sous une forme plus mathématique. Chaque aménagement de la maison est représenté par une lettre : t, la terrasse ; e, l'étage ; p, le patio ; g, le garage. Chacune correspond à une variable logique qui prend deux valeurs : V (vrai) lorsque l'option est retenue ou F (faux) quand l'option est rejetée. La première contrainte (il faut qu'il y ait un étage ou une terrasse, mais pas les deux) est satisfaite lorsque e ou t sont vrais, mais pas e et t à la fois. Cette distinction n'est pas contenue dans le «ou» logique, noté «v». Aussi, la proposition «(*non e*) ou (*non t*)» – soit la maison n'a pas d'étage, soit elle

1. LES SOUHAITS ET LES CONTRAINTES d'un particulier pour la construction d'une maison constituent les clauses d'un problème SAT que les informaticiens cherchent à résoudre.

de la complexité calculatoire!

n'a pas de terrasse – doit également être satisfaite. On écrit symboliquement «$\neg e$» la négation de la variable e. La première contrainte se réécrit donc sous forme de deux propositions élémentaires, nommées clauses, $e \vee t$ et $\neg e \vee \neg t$, qui doivent être satisfaites. Ainsi, chaque terme d'une clause est une variable logique ou sa négation. Les trois autres contraintes de notre exemple s'écrivent de la même façon sous la forme de clauses. Au total, on obtient sept clauses : $e \vee t$, $\neg e \vee \neg t$, $p \vee g$, $\neg p \vee \neg g$, $\neg e \vee t \vee \neg p$, $\neg t \vee \neg g$ et $e \vee \neg g$.

L'ensemble de ces clauses, qui représente les contraintes du client, est la formule SAT du problème. Le problème consiste à attribuer une valeur V ou F à chaque variable de façon à ce que toutes les clauses de la formule soient satisfaites. Quand c'est possible, la formule est satisfaisable ; dans le cas contraire, la formule est insatisfaisable. Notre client est chanceux, sa formule SAT est satisfaisable de deux façons : e est faux, t est vrai, p est vrai et g est faux, ou bien, e est vrai, t est faux, p est faux et g est vrai. En d'autres termes, une maison autorisée est de plain-pied, dotée d'une terrasse et d'un patio, mais n'a pas de garage ; l'autre a un étage et un garage, mais n'a ni terrasse ni patio.

Complexité de résolution

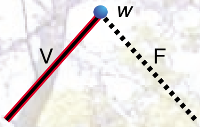
La difficulté à résoudre un tel problème dépend principalement du nombre de clauses et du nombre de variables. Notre exemple, avec sept clauses et quatre variables est facilement soluble. Toutefois, la difficulté croît rapidement, car on ne dispose pas de méthode générale de résolution beaucoup plus efficace que l'énumération naïve de toutes les combinaisons jusqu'à en trouver une qui convienne ou constater qu'aucune ne convient. Une seule option supplémentaire, telle la possibilité pour notre acheteur immobilier d'avoir une grille en fer forgé, double le nombre de combinaisons à énumérer et par conséquent double le temps de résolution.

Quand le nombre de clauses et le nombre de variables sont grands, l'ordinateur devient indispensable. La taille d'une formule SAT se mesure par le nombre de variables, soit quatre pour notre exemple. Pour le résoudre, on énumère au maximum 2^4, soit 16, combinaisons. L'énumération est de 2^n combinaisons pour une formule SAT à n variables. Ainsi, le temps d'énumération des combinaisons d'une formule SAT est une fonction exponentielle du nombre de variables. Pour un petit nombre de variables, c'est acceptable, mais ce temps croît rapidement avec la taille du problème. Comparons le temps de calcul

L'ALGORITHME DE DAVIS, PUTNAM ET LOVELAND

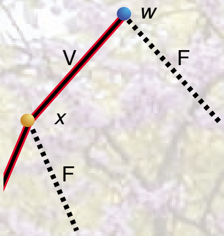
$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

1. À partir d'une formule SAT *(dans le cadre jaune)*, on construit peu à peu l'arbre des combinaisons où chaque nœud représente une variable *(w, x, y et z)* et les branches, le choix d'une valeur (*Vrai* ou *Faux*) pour cette variable.



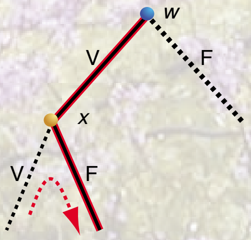
$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

2. L'algorithme attribue la valeur *Vrai (représenté par le trait rouge)* à la variable *w* afin de satisfaire la première clause. Puis, pendant la phase de propagation, il simplifie la formule SAT *(dans le cadre bleu)* : les clauses satisfaites et les négations de la variable *w* sont supprimées *(en blanc)*. Ainsi, l'algorithme obtient une nouvelle formule SAT *(en noir)*



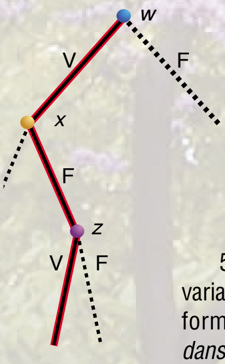
$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

3. L'algorithme attribue la valeur *Vrai* à la variable *x* afin de satisfaire la deuxième clause. Toutefois, la phase de propagation aboutit à une contradiction (*en rouge dans le cadre bleu*).



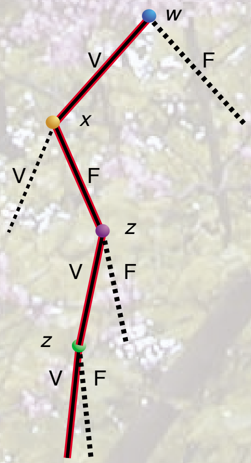
$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

4. L'algorithme remonte dans l'arbre *(flèche rouge)* et attribue l'autre valeur *(Faux)* à la variable *x*. Une nouvelle phase de propagation simplifie la formule SAT *(en noir, dans le cadre bleu)*.



$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

5. L'algorithme attribue la valeur *Vrai* à la variable *z*. La phase de propagation réduit la formule SAT à une clause unitaire *(en noir, dans le cadre bleu)* qui impose le choix de la dernière valeur.



$w \vee \neg x \vee y$
$\neg w \vee x \vee z$
$\neg w \vee \neg x \vee \neg y$
$\neg w \vee \neg x \vee y$
$x \vee y \vee \neg z$

6. L'algorithme attribue la valeur *Vrai* à la variable *y*. Toutes les clauses de la formule SAT sont satisfaites : *w* est *Vrai*, *x Faux*, *z Vrai* et *y Vrai*.

L'algorithme consiste en une succession d'essais et de corrections d'erreurs qui peu à peu dessinent un arbre. À chaque étape, l'algorithme attribue une valeur à la première variable de la première clause de façon à satisfaire cette dernière. Puis, il analyse les conséquences logiques de son choix lors d'une phase de propagation : les clauses satisfaites sont supprimées, alors que les clauses où apparaît la négation de la variable sont réduites aux autres variables, qui seules pourront satisfaire la clause. L'algorithme réitère la même opération jusqu'à découvrir une solution, ou bien, en l'absence de solution, montrer que la formule SAT est insatisfaisable.

d'un algorithme de complexité polynomiale (par exemple le temps de résolution est égal au cube du nombre de variables), avec celui d'un algorithme de complexité exponentielle, sachant qu'un ordinateur classique effectue une opération élémentaire en une microseconde. Pour une formule à 10 variables, les deux algorithmes énumèrent toutes les combinaisons en environ une milliseconde. Avec 50 variables, le premier opère en 125 millisecondes environ, alors que le second requiert 35 ans. Pire encore, avec 100 variables, le premier met une seconde là où le second met plus de 400 000 milliards de siècles !

Hélas, le problème SAT est le plus souvent dans la seconde situation, car on ne dispose pas d'algorithme de complexité polynomiale pour le résoudre bien que personne n'ait encore prouvé qu'il n'en existe pas. Cependant, le consensus est d'admettre qu'il n'en existe pas. En effet, on a établi un lien entre le problème SAT et de multiples autres problèmes en mathématiques, en logique, en informatique et même en physique théorique et en biologie qui se trouvent dans la même situation : tous ont la même complexité, de sorte qu'un algorithme de complexité polynomiale découvert pour l'un d'entre eux serait applicable pour chacun des autres. Ces problèmes sont dits NP-complets *(voir l'encadré page xx).*

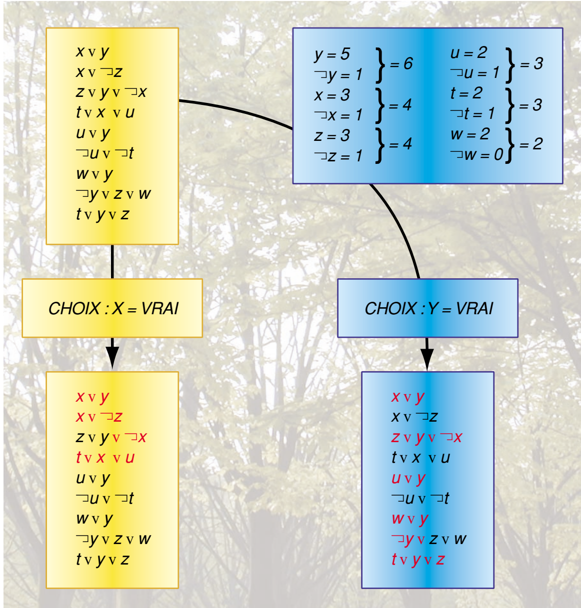
Un arbre de recherche

Les algorithmes les plus efficaces pour résoudre le problème SAT sont fondés sur une procédure dite de Davis, Putnam et Loveland, du nom des mathématiciens qui l'ont élaborée en 1962. Elle consiste en une sorte de cheminement, par succession d'essais et de corrections d'erreurs, que l'on représente sous la forme d'un arbre de recherche qui croît à mesure que l'on résout la formule SAT *(voir l'encadré de la page xx).* Dans un tel arbre, chaque nœud représente une variable et chaque branche le choix d'une valeur pour la variable associée au nœud d'où elle est issue. L'algorithme choisit une variable dans la première clause du problème, par exemple *e* dans le cas de l'architecte, et lui attribue une valeur logique afin de satisfaire la clause, ici *Vrai*. Ensuite, pendant une phase dite de propagation, l'algorithme analyse les conséquences logiques du choix effectué : les clauses où la variable choisie n'apparaît pas ne sont pas modifiées ; les clauses où apparaît cette variable sont éliminées, car satisfaites ; enfin, celles où figure la négation de la variable sont réduites aux autres variables qui restent seules en lice pour satisfaire la clause. L'algorithme répète la même opération sur la nouvelle formule SAT.

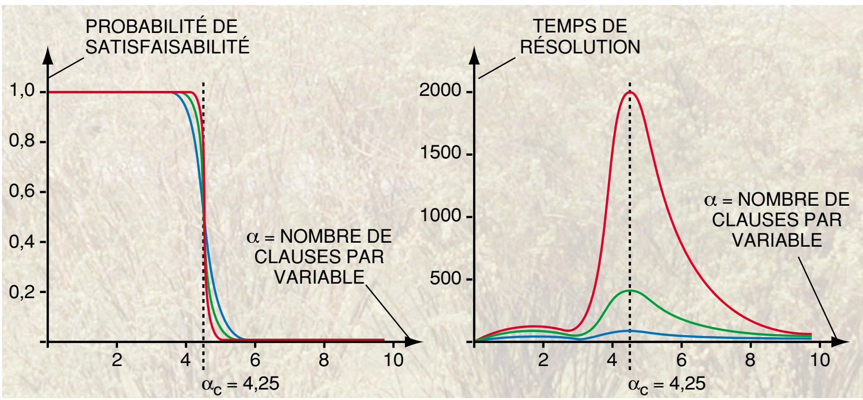
Lorsque l'algorithme aboutit à deux clauses unitaires contradictoires, par exemple x et $\neg x$, il «remonte» dans l'arbre au nœud précédent et modifie la valeur de la dernière variable pour explorer une nouvelle branche. Selon que la formule SAT est satisfaisable ou non, une solution est atteinte ou toutes les branches se terminent par une contradiction.

Bien que rudimentaire, l'algorithme de Davis, Putnam et Loveland, est le meilleur procédé connu pour résoudre des formules SAT. Il est la base d'algorithmes plus élaborés qui utilisent des «heuristiques» pour guider le parcours dans l'arbre de recherche *(voir la figure 2).* Ces heuristiques choisissent à chaque nœud de l'arbre une variable et lui attribuent une valeur de sorte que l'algorithme soit s'oriente le plus rapidement vers une solution, soit détecte dès que possible qu'il n'y a pas de solution. Les heuristiques influent parfois notablement sur le temps de résolution d'une formule SAT. Nous verrons que grâce aux études de physique statistique, nous avons mis au point des heuristiques très efficaces pour résoudre des formules SAT difficiles.

Par exemple, une des heuristiques peut être de repérer, d'une part, qu'une variable x est répétée souvent dans les clauses et, d'autre part, que cette variable est plus fréquente sous une forme positive (x) que négative ($\neg x$) : l'algorithme lui attribuera préférentiellement la valeur *Vrai* afin d'augmenter les chances de trouver



2. LES HEURISTIQUES sont des astuces informatiques qui accélèrent la découverte d'une solution satisfaisant une formule SAT *(en haut, à gauche)*. Un algorithme dépourvu d'heuristique attribue une valeur (*Vrai*) à la première variable (*x*) de façon à satisfaire la clause. La phase de propagation simplifie la formule qui contient six clauses *(en bas, à gauche)*. Une heuristique *(en haut, à droite)* peut repérer la variable la plus souvent répétée (*y*) et reconnaître qu'elle est plus souvent sous sa forme positive : il attribue alors la valeur *Vrai* à cette variable. La phase de propagation réduit la formule à quatre clauses seulement *(en bas, à droite)*.



3. LE PHÉNOMÈNE DE SEUIL *(à gauche)* apparaît lorsqu'on étudie la satisfaisabilité de formules SAT aléatoires. Lorsque α, le nombre moyen de clauses par variable est inférieur à une valeur critique α_c = 4,25, toutes les formules SAT sont satisfaisables. À l'inverse, quand α est supérieur à la valeur critique, toutes les formules SAT sont insatisfaisables. Le passage d'un état à l'autre est d'autant plus marqué que le nombre de variables est élevé *(50 variables en bleu, 75 variables en vert et 100 variables en rouge)*. La mesure du temps de résolution *(à droite)*, estimé à partir de la taille de l'arbre de recherche, c'est-à-dire le nombre de nœuds qu'il contient, montre que plus le nombre moyen de clauses par variable est proche de la valeur critique, plus le temps de résolution est long.

À la conquête du seuil

Les mathématiciens essaient de calculer précisément la valeur du seuil de transition α_c rencontré dans la résolution du problème 3-SAT afin de donner une assise rigoureuse à un phénomène observé «expérimentalement». Ce calcul est difficile, aussi tentent-ils de cerner cette valeur dans un intervalle de plus en plus réduit, en calculant, par des méthodes différentes, les bornes inférieures et supérieures de cet intervalle.

La voie exploitée pour calculer une borne inférieure du seuil est l'analyse de la probabilité d'obtenir une solution par un algorithme de type Davis, Putnam et Loveland simplifié. Par exemple, quand le rapport α est inférieur à 1,63, un algorithme qui attribue une valeur uniquement aux variables pures (les variables qui, dans une formule SAT aléatoire, apparaissent toujours sans négation ou toujours avec une négation) de façon à satisfaire ces variables, trouve une solution avec une probabilité de quasiment 100 pour cent. Cette valeur de 1,63 était la première borne inférieure connue. Récemment, grâce à un algorithme qui explore une branche d'un arbre de recherche et autorise un nombre limité de remontées dans l'arbre face à une contradiction, cette borne inférieure a été déplacée à 3,26. C'est la borne inférieure la plus élevée

L'établissement de bornes supérieures diffère totalement. Plutôt que la probabilité de satisfaisabilité, on calcule le nombre moyen de solutions par formule : quand le nombre moyen de solutions d'une formule 3-SAT aléatoire ayant un rapport α donné tend vers zéro lorsque le nombre de variables n augmente indéfiniment, alors la probabilité qu'il y ait au moins une solution tend vers zéro pour cette valeur de α. Cet argument donne une première borne supérieure égale à 5,19. Pourquoi diffère-t-elle autant de la valeur attendue 4,25? Parce qu'entre 5,19 et le seuil, quelques rares formules satisfaisables ont un très grand nombre de solutions : ainsi, le nombre moyen de solutions est élevé bien que la plupart des formules n'ont pas de solution ; on est alors décalé par rapport au seuil réel. L'impact sur la moyenne de l'abondance de solutions dans cette minorité de formules diminue quand, par exemple, on compte pour une seule des solutions similaires selon un critère donné. Une borne supérieure égale à 4,643 a été ainsi obtenue.

Toutefois, agir sur les solutions de la minorité de formules indésirables a vite atteint ses limites. Les progrès suivants ont montré qu'il valait mieux négliger ces formules indésirables dans le calcul de la moyenne. En effet, on constate expérimentalement qu'un programme qui crée des formules 3-SAT aléatoires pour un rapport α légèrement au-dessus du seuil ne produit jamais une de ces formules avec beaucoup de solutions. Ces formules sont «atypiques». Aussi des critères de «typicité» ont été définis afin d'éliminer du calcul les formules «atypiques» : par exemple, la distribution de la fréquence d'apparition d'une variable dans une formule SAT aléatoire (celles qui apparaissent une fois, celles qui apparaissent une fois sans négation et une fois avec négation…). Nous avons combiné cette approche avec la précédente et calculé une nouvelle borne supérieure égale à 4,506. C'est aujourd'hui la borne supérieure la plus basse.

une solution. Ainsi dotés de l'énoncé d'un problème et d'un algorithme de résolution, étudions une curiosité du problème SAT.

Au seuil de la maison

Tout problème SAT dont chaque clause contient exactement 2 variables, ou problème 2-SAT, a une complexité polynomiale qui est une fonction du carré du nombre de variables. En revanche, tous les algorithmes connus pour un problème 3-SAT ont une complexité exponentielle. Le saut de complexité entre le problème 2-SAT et le problème 3-SAT en a fait des objets d'étude expérimentaux et théoriques les plus importants. L'origine et la compréhension de ce saut dans la difficulté de résolution des problèmes SAT sont éclairées par l'existence d'un phénomène de seuil, un comportement qui a suscité l'étonnement au moment de sa découverte en 1991.

On crée aléatoirement des formules SAT, nommées formules SAT aléatoires, en maintenant constant le rapport α du nombre de clauses sur le nombre de variables. Plus α est petit, plus une formule est satisfaisable. Un client avec beaucoup d'options et peu de contraintes a facilement une maison. À l'inverse, plus α est grand, plus une formule est insatisfaisable. Avec beaucoup de contraintes et peu de variables, notre client a toutes les chances de rester locataire.

Les expérimentations informatiques mettent en évidence un petit intervalle autour d'une valeur critique, nommée α_c, qui sépare deux zones : en deçà de α_c, presque toutes les formules sont satisfaisables ; au-delà, presque toutes les formules sont insatisfaisables. L'intervalle de transition est d'autant plus petit que la taille des formules, c'est-à-dire le nombre de variables et le nombre de clauses augmentent *(voir la figure 3)*. Pour des formules infiniment grandes, l'intervalle se réduit au point α_c. Pour les problèmes 2-SAT, α_c est égal à un. Pour les problèmes 3-SAT, α_c est égal à environ 4,25 : c'était la première fois qu'on découvrait expérimentalement un phénomène de seuil pour un problème NP-complet! Un défi était lancé aux mathématiciens.

Au regard de la complexité de résolution, ce phénomène de seuil présentait un intérêt notable. En effet, le temps moyen pour résoudre une formule 3-SAT aléatoire (montrer qu'elle est satisfaisable ou non) est d'autant plus long que α est proche de α_c *(voir la figure 3)*. En d'autres termes, les formules les plus difficiles à résoudre sont concentrées autour du seuil. Ainsi, l'identification de ce seuil a circonscrit des formules SAT très difficiles à résoudre. Par ailleurs, ce seuil oriente l'étude des problèmes SAT : quand pour une formule 3-SAT, α est inférieur à α_c, on cherche surtout à en trouver une solution ; à l'inverse, quand α est supérieur à α_c, on cherche surtout à montrer que la formule est insatisfaisable.

Ces observations expérimentales soulèvent plusieurs questions. Comment démontrer rigoureusement l'existence d'une valeur critique α_c où a lieu la transition et calculer cette valeur? En quoi les formules difficiles au niveau de la transition se distinguent-elles des autres formules? Comment en améliorer l'efficacité de résolution?

Encadrement du seuil

Grâce à un lien étroit entre les formules 2-SAT et la théorie des graphes, au début des années 1990, plusieurs mathématiciens ont montré l'existence d'une transition de phase dans les problèmes 2-SAT ; ils ont établi que la valeur du seuil de transition est égale à un. La situation des problèmes 3-SAT est bien différente! Le calcul de la probabilité de satisfaisabilité d'une formule 3-SAT pour déterminer directement le seuil de transition du problème 3-SAT est vite apparu très difficile ; il est même toujours hors de portée. Aussi, les informaticiens se sont orientés vers une approche indirecte du calcul du seuil en délimitant un intervalle de plus en plus petit autour de sa valeur *(voir l'encadré page xx)*. Pour ce faire, on cherche, d'une part, des bornes inférieures de l'intervalle les plus élevées possibles en deçà desquelles presque toutes les formules sont satisfaisables et, d'autre part, des bornes supérieures de l'intervalle, les plus faibles possibles au-delà desquelles presque toutes les formules sont insatisfaisables. On divise ainsi la difficulté par deux puisqu'on se consacre indépendamment, et avec des méthodes différentes, à l'une des deux zones où les solutions sont facilement accessibles. Depuis une dizaine d'années, l'étau se resserre autour du seuil pour les problèmes 3-SAT : aujourd'hui, la meilleure borne inférieure est égale à 3,260, et la meilleure borne supérieure à 4,506. Toutefois, si la valeur du seuil se précisait, on ignorait encore les causes de ce phénomène de seuil.

Le phénomène de seuil décrit rappelle les transitions de phases observées en physique de la matière condensée. L'analogie est plus profonde qu'il n'y paraît. Ainsi, dans les systèmes physiques, tel un gaz, les grandeurs macroscopiques, par exemple la pression, résultent de phénomènes microscopiques, le choc des molécules contre les parois. De la même façon, la satisfaisabilité ou l'insatisfaisabilité d'une formule SAT est une propriété globale qui résulte de la structure détaillée de la multitude des clauses qui la composent, et qui varie en fonction de paramètres externes, ici α. Prédire le comportement statistique global d'un système à partir de la connaissance de sa structure à un niveau élémentaire est précisément la tâche assignée à la physique statistique.

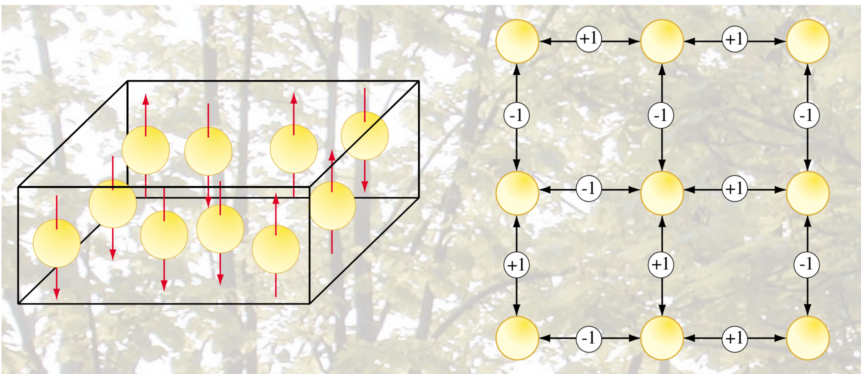
En premier lieu, traduisons le problème SAT en une formulation plus familière aux physiciens statisticiens. Par analogie avec les spins, c'est-à-dire les moments cinétiques de particules, à chaque variable x, tels un garage ou une terrasse, est associée une variable de spin S_x égale à +1 quand x est vraie et égale à -1 dans le cas contraire. Chacune des 2^n énumérations des valeurs des n variables est une configuration des spins. On associe à chaque configuration un coût dont la valeur reflète le nombre de clauses non satisfaites par cette configuration. Il s'agit de déterminer la configuration de coût minimal, ou configuration fondamentale : une valeur nulle ou positive du coût minimal indique alors une formule 3-SAT respectivement satisfaisable ou non.

Le coût correspondant à une formule 3-SAT aléatoire est souvent une fonction compliquée des spins. Des fonctions similaires, dépendant elles aussi d'un grand nombre de variables de spins, sont fréquentes dans la modélisation théorique de certains matériaux magnétiques découverts au milieu des années 1970, les verres de spins *(voir la figure 4)*. Ces derniers sont des alliages composés d'une matrice métallique où sont intercalés de façon désordonnée des ions magnétiques. Chaque ion est doté d'un moment magnétique S, nommé spin, qui s'oriente selon une seule direction, l'axe vertical, imposée par la matrice. Le moment magnétique S prend deux valeurs : $S=+1$ quand le spin pointe vers le haut, $S=-1$ lorsqu'il est orienté vers le bas. Dans un matériau magnétique classique, les ions occupent des positions régulières et leurs spins agissent les uns sur les autres de façon cohérente, on parle alors de couplage positif. L'énergie d'interaction d'un tel matériau, c'est-à-dire la somme des interactions des paires de spins, est favorable lorsque tous les spins sont orientés dans la même direction. En revanche, dans un verre de spin où les ions magnétiques sont répartis aléatoirement, les interactions magnétiques alignent les spins correspondants, soit dans le même sens, c'est un couplage positif, soit dans des sens opposés dans le cas d'un couplage négatif.

Trouver le minimum énergétique pour un ensemble d'interactions donné est souvent ardu, notamment à cause de ces couplages positifs et négatifs qui incitent un même spin à pointer dans des directions opposées. Les configurations fondamentales, d'énergie nulle, sont multiples. Chacune d'entre elles est désordonnée : certains spins pointent vers le haut, d'autres vers le bas.

Verres de spins et problème SAT

Comment l'énergie d'un verre de spin se compare-t-elle à la fonction de coût d'une formule SAT, telle celle de notre architecte? La clause $e \vee \neg g$ est contredite quand e est fausse et g vraie ; sinon, elle est satisfaite. Le coût associé à cette clause s'écrit sous la forme $(1-S_e)(1+S_g)/4$: en effet, ce produit vaut un quand $S_e=-1$ et $S_g=+1$, zéro sinon. Le coût total de la formule SAT de l'architecte est la somme des coûts individuels des clauses. Lorsqu'on développe cette expression, on obtient un polynome qui contient des termes en S_x, des termes en S_xS_y et un terme en $S_xS_yS_z$ où x, y et z sont des variables quelconques de la



4. UN VERRE DE SPINS *(à gauche)* est composé d'une matrice métallique régulière où des ions magnétiques *(en jaune)* sont intercalés de façon aléatoire. Le spin magnétique *(flèche rouge)* de ces ions est orienté vers le haut ou vers le bas. On modélise un tel matériau *(à droite)* en représentant les interactions, ou couplages, des paires de spins : un couplage positif (+1) tend à aligner les spins des ions associés dans le même sens ; un couplage négatif (-1) les aligne dans des sens opposés. On cherche ensuite le minimum énergétique du matériau en tenant compte de ces interactions. Une formule SAT est parfois traduite en une fonction de coût, similaire au modèle des interactions dans un verre de spin, que l'on cherche à minimiser afin de vérifier si la formule SAT est satisfaisable (le coût est nul) ou non (le coût est supérieur à zéro).

formule. Le terme constant du polynome ne joue aucun rôle dans la minimisation de la fonction de coût. Les termes à deux et trois variables de spins révèlent les influences qu'ont les variables associées entre elles : la présence de deux variables de spin dans un même terme indique que la valeur de l'une des variables associées de la formule SAT a une influence sur la valeur de l'autre. Ainsi, on modélise un système mathématique en des termes propres à la physique. Avec le modèle obtenu, on étudie le comportement global du système et l'on souhaite déterminer la valeur du seuil critique de la transition de phase. Il s'agit de comprendre comment les variables interagissent et comment les unes dictent leur loi aux autres.

Quand on refroidit un verre de spin, son énergie diminue. En deçà d'une température critique, les propriétés magnétiques changent brutalement. Nous retrouvons l'idée qu'au-delà d'une valeur α_c (l'équivalent de l'inverse de la température du verre de spins), toutes les formules 3-SAT cessent d'être satisfaisables. Quand α augmente, les formules passent d'un état satisfiable à un état insatisfaisable ; quand la température diminue, le verre de spins passe d'un état non magnétique (la valeur des spins est égale indifféremment +1 ou -1) à un état magnétique désordonné (chaque spin a une valeur préférée). Des calculs sur le modèle physique ont montré que α_c est égal à 4,36, soit environ 2,5 pour cent supérieur à la valeur numérique issue des expérimentations informatiques. Cette différence résulte d'approximations au cours de l'analyse non rigoureuse du modèle physique. Cependant, l'étude physique du modèle de verre de spins est riche d'enseignements, notamment elle élucide pourquoi certaines formules SAT sont difficiles à résoudre, alors que les mathématiques les avaient seulement localisées.

Quand les variables gèlent

Quand une formule SAT est facile à résoudre (le verre de spins correspondant a une température supérieure à la température critique), l'algorithme a le choix quant à la valeur qu'il attribue aux variables, et la formule SAT a de nombreuses solutions. Dans notre exemple, toutes les variables ont des valeurs différentes dans les deux solutions de l'architecte. Le franchissement du seuil, suite à l'ajout d'un nombre de clauses très petit devant le nombre de variables s'accompagne de l'apparition brutale de nombreuses variables contraintes, ou gelées, c'est-à-dire qui ont toujours les mêmes valeurs quelles que soient les solutions. Dans notre exemple, ce serait l'obligation d'avoir un patio quelles que soient les maisons possibles.

La proportion des variables gelées est nulle au-dessous du seuil et passe, de façon discontinue, à environ 15 pour cent au-delà de α_c. La transition de 3-SAT s'apparente aux transitions dites du premier ordre, caractérisée par un passage brusque et discontinu d'un état à un autre, telle la transition de l'eau liquide en glace où le volume occupé par une masse d'eau liquide donnée augmente brutalement lors du passage à la phase solide.

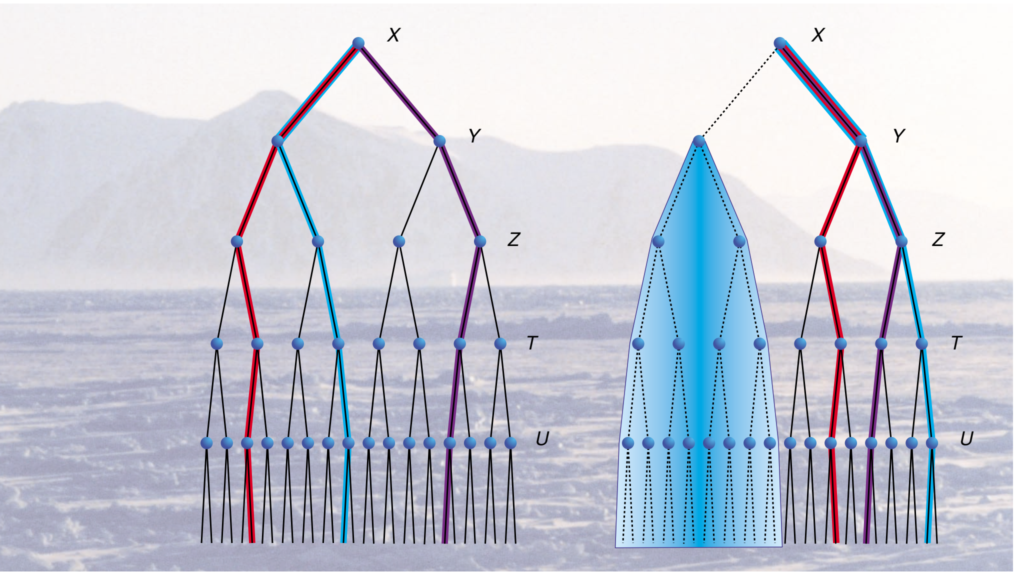
L'apparition de variables gelées montrait enfin pourquoi les formules SAT sont difficiles à résoudre au seuil. En effet, lorsque l'algorithme explore l'arbre des combinaisons, il affecte une valeur à une variable. Quand celle-ci n'est pas gelée, il trouvera vraisemblablement une solution. En revanche, lorsque cette variable est gelée (et que l'algorithme lui attribue la mauvaise valeur), l'algorithme perd beaucoup de temps à attribuer des valeurs aux variables suivantes sans obtenir de solution : il lui faut remonter jusqu'à la variable gelée et lui attribuer l'autre valeur *(voir la figure 5)*.

Trajectoires de résolution

Comment l'existence de la transition influe-t-elle sur la complexité de résolution ? Lorsque l'algorithme de Davis, Putnam et Loveland opère, non seulement les nombres de variables et de clauses de la formule SAT initiale sont modifiés, et donc le paramètre α, mais aussi la nature des clauses. En simplifiant certaines clauses, l'algorithme transforme la formule 3-SAT initiale en une formule mixte qui contient des clauses à deux ou à trois variables : la formule est alors 2+p-SAT où p représente la proportion des clauses à trois variables, au départ égale à un, et qui varie entre 0 et 1 au cours de la résolution.

À l'instar du problème 3-SAT, les formules 2+p-SAT sont presque toujours soit satisfaisables, soit insatisfaisables, selon que α est inférieur ou supérieur à un seuil $\alpha_{c(p)}$ qui dépend de p. En termes physiques, on représente le diagramme de phase de 2+p-SAT dans un plan, avec p en abscisse et α en ordonnées *(voir la figure 6)* : la ligne critique définit la frontière entre la phase satisfaisable (au-dessous de la ligne) et la phase insatisfaisable (au-dessus). Un tel diagramme de phases à deux paramètres est commun en physique, par exemple celui de l'eau où, selon la température et la pression, on distingue les phases gazeuse, liquide et solide.

Sous l'action de l'algorithme, les paramètres p et α de la formule évoluent. Le diagramme montre l'évolution du point de coordonnées α et p, caractéristique de la formule. Selon que la trajectoire du point reste confinée



5. LES VARIABLES GELÉES (des variables qui ont la même valeur quelle que soit la solution) augmentent le temps de résolution d'une formule SAT par l'algorithme de Davis, Putnam et Loveland. Dans le cas d'une formule SAT satisfaisable, quand aucune variable n'est gelée *(à gauche)*, l'algorithme trouve une solution *(en rouge, en violet et en bleu)* facilement. En revanche, quand par exemple la variable *x* est gelée *(à droite)*, et qu'il attribue à cette variable la mauvaise valeur, il explore une grande partie de l'arbre *(en bleu)* sans trouver de solution.

Les problèmes NP-complets et les phénomènes de seuil

Le problème *k*-SAT, avec *k* supérieur ou égal à trois, est un problème NP (abréviation de Non déterministe Polynomial), c'est-à-dire qu'il pourrait être résolu en un temps polynomial par un ordinateur «non déterministe» ayant la possibilité de «deviner» la ou une solution : il se contenterait de vérifier qu'il s'agit bien d'une solution. Cette définition théorique englobe de multiples problèmes dont beaucoup ont un intérêt pratique, mais pour lesquels on n'a pas d'algorithmes de complexité polynomiale : on ne dispose pour ces problèmes que d'algorithmes de complexité exponentielle. En revanche, parmi les problèmes NP, ceux qui peuvent être résolus par un algorithme de complexité polynomiale forment la classe des problèmes P. Les problèmes NP sont au centre des préoccupations des mathématiciens, car l'impression que la classe des problèmes P est strictement incluse dans celle des problèmes NP n'a jamais été démontrée. Cette conjecture se résume à la question P=NP?

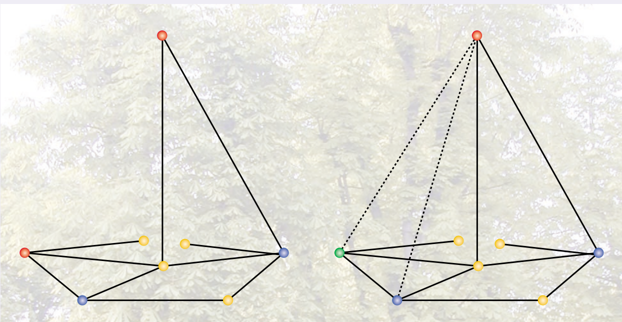
En 1971, le mathématicien canadien Stephen Cook a prouvé que l'existence d'un algorithme résolvant le problème SAT en un temps polynomial entraînait que tous les problèmes NP seraient également solubles en un temps polynomial. En conséquence, le problème SAT a été nommé NP-complet. On connaît aujourd'hui plusieurs milliers de problèmes qui ont la même propriété : un algorithme de complexité polynomiale pour un seul d'entre eux serait applicable immédiatement pour chacun d'eux. Une telle découverte apporterait une réponse positive à la question P=NP?, en d'autres termes, on résoudrait tous les problèmes par un algorithme à complexité polynomiale. À l'inverse, une réponse négative confirmerait ce que laissent subodorer les tentatives jusqu'à présent infructueuses sur les problèmes SAT ou celui du voyageur de commerce. Cette énigme figure sur la liste des sept problèmes majeurs du troisième millénaire désignés en mai 2000. Pour chacun d'entre eux, la solution trouvée sera récompensée par un million de dollars par l'Institut Clay.

Le coloriage de graphe est un autre problème NP-complet, similaire au problème *k*-SAT. Dans les atlas, les pays qui ont une frontière commune se distinguent, car on leur attribue des couleurs différentes. Le problème du coloriage d'un graphe est une généralisation de cet exemple : il s'agit de colorier les sommets d'un graphe de façon à ce que deux sommets reliés par une arête n'aient jamais la même couleur *(voir la figure à gauche)*. La question est de déterminer le nombre minimal *k* de couleurs nécessaires pour colorier un graphe donné.

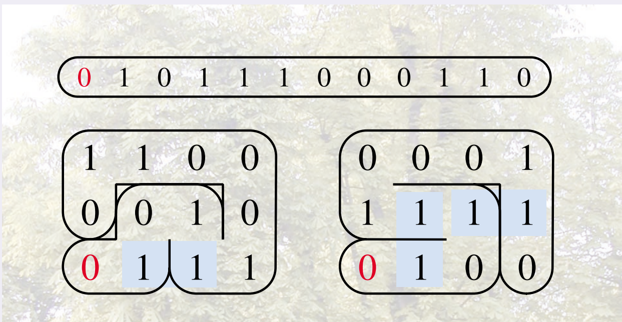
Le problème du *k*-coloriage donne lieu à des phénomènes de seuil. Il a fait l'objet d'études combinatoires inspirées des techniques élaborées pour le problème *k*-SAT. Les graphes à colorier sont créés aléatoirement à partir d'un ensemble de *n* sommets et caractérisés par un seul paramètre, le rapport *C* du nombre d'arêtes sur le nombre de sommets. Dès 1960, Paul Erdös a posé la question de l'existence d'une valeur critique C_k au-dessous de laquelle, lorsque *n* tend vers l'infini, tous les graphes sont *k*-coloriables, et au-dessus de laquelle presque aucun ne l'est. Les expérimentations numériques indiquent un seuil autour de C_k environ égal à 2,3 pour *k* égal à trois. Les meilleures bornes disponibles sont obtenues par des méthodes similaires à celles employées pour le problème *k*-SAT : C_k serait compris entre 1,923 et 2,4682.

La partition des nombres, c'est-à-dire comment répartir un ensemble d'entiers de façon à former deux ensembles dont les sommes sont égales, est aussi un problème NP-complet qui présente des phénomènes de seuil. La taille de l'expression formalisée du problème n'est pas ici le nombre d'entiers *n*, mais le produit de celui-ci par la taille *b* (en nombre de bits) du plus grand d'entre eux, ceci afin de tenir compte de la taille des entiers eux-mêmes. Quand les entiers sont tirés aléatoirement de façon à maintenir constant le rapport $k=b/n$, les expérimentations numériques indiquent que, lorsque *n* devient infiniment grand, une partition parfaite existe presque toujours quand *k* est inférieur à une valeur critique k_c et est presque toujours impossible quand *k* est supérieur à cette valeur critique. Estimée empiriquement à 0,96, la valeur de k_c a été calculée précisément, en 1998, par le physicien Stephan Mertens, de l'Université Otto von Guericke, à l'aide de calculs de physique statistique : elle est égale à 1.

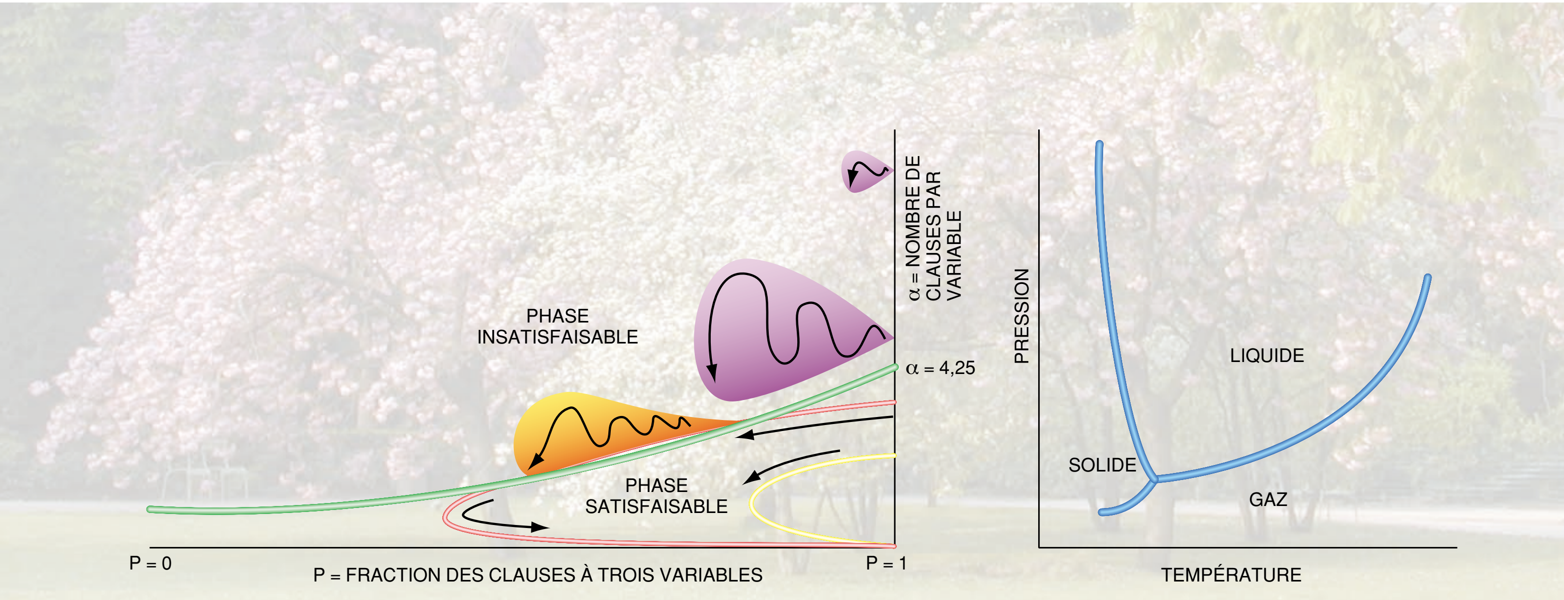
L'informatique n'est pas le domaine réservé des problèmes NP-complets. Ainsi, on en connaît en logique, telle la recherche d'un ensemble minimum d'axiomes pour une théorie logique, et en physique, par exemple, la recherche de l'état de plus basse énergie d'un verre de spin, un alliage de métal et d'ions magnétiques. Plus surprenant, les problèmes NP-complets se rencontrent aussi en biologie. Le problème du repliement des protéines, c'est-à-dire de la prédiction de leur structure tridimensionnelle à partir de leur séquence en acides aminés, est une étape importante pour comprendre le passage des gènes aux protéines. Une protéine dans un organisme vivant ne peut exister que sous une seule des configurations géométriques a priori possibles (à l'exception des protéines prions). Cette configuration protéique minimise l'énergie totale de la molécule. Dans un modèle schématisé à deux dimensions *(voir la figure à droite)*, la protéine est une suite de 1 et de 0 qui symbolisent des chaînes hydrophiles et neutres. Il s'agit d'inscrire cette suite dans une grille plane de sorte que le nombre de 1 adjacents soit maximal. On a montré récemment que cette version simplifiée équivaut à un problème NP-complet.



Le problème du *k*-coloriage consiste à colorier les sommets d'un graphe avec *k* couleurs de sorte que deux sommets joints par une arête aient une couleur différente. Par exemple, trois couleurs suffisent à colorier le graphe de gauche, alors que deux arêtes supplémentaires *(en pointillés)* obligent l'ajout d'une quatrième couleur *(à droite)*.



Une protéine *(en haut)* adopte une configuration qui minimise son énergie. Ainsi, les chaînes hydrophiles *(1)* ont tendance à se rapprocher, alors que celles qui sont neutres *(0)* n'interviennent pas. Prédire le repliement consiste à déterminer la structure où le nombre de 1 adjacents *(cernés de bleu)* est maximal *(à droite)*.



6. LE DIAGRAMME DE PHASE du problème 2+*p*-SAT *(à gauche)* représente la satisfaisabilité en fonction de *p* et de α. On distingue deux régions (satisfaisable et insatisfaisable) séparée par une ligne critique *(en vert)*. Certaines formules SAT *(en jaune)* restent en-deçà de cette ligne lors de la résolution par l'algorithme de Davis, Putnam et Loveland qui trouve facilement une solution. D'autres *(en violet)* restent confinées dans la région insatisfaisable : l'algorithme explore tout l'arbre des combinaisons sans découvrir de solution. Enfin, quelques formules SAT *(en rouge)* sont initialement dans la région satisfaisables, puis coupe la ligne critique pendant un temps où l'algorithme explore systématiquement une région de l'arbre *(zone orange)* avant de retourner dans la région satisfaisable pour trouver une solution. Ce diagramme de phases est similaire à celui de l'eau *(à droite)* où selon la température et la pression, elle est dans une phase liquide, solide ou gazeuse.

dans la phase satisfaisable, dans la phase insatisfaisable ou traverse la frontière entre les deux, la complexité de résolution change drastiquement. Les paramètres *p* et α définissent une frontière qui sépare les problèmes SAT à complexité polynomiale des problèmes SAT à complexité exponentielle. On distingue trois intervalles successifs selon α initial. Pour une valeur α inférieure à α_L environ égal à trois, la trajectoire reste confinée à la phase satisfaisable : la résolution, rapide, a ici une complexité fonction du nombre de variables. Résoudre un problème 3-SAT est facile dans cet intervalle. À l'inverse, dans la phase non satisfaisable, quand α est supérieur à α_c environ égal à 4,3, les problèmes 3-SAT n'ont (presque) jamais de solution, mais l'algorithme met un temps considérable avant de s'en assurer. La trajectoire correspondante ne s'échappe pas de la phase non satisfaisable et occupe toute une portion du diagramme de phase, c'est-à-dire que l'algorithme essaie de nombreuses énumérations. La complexité croît alors exponentiellement avec le nombre de variables. Enfin, la région intermédiaire $\alpha_L<\alpha<\alpha_C$ est la plus riche, car elle conjugue les deux comportements précédents. La formule 3-SAT initiale est dans la phase satisfaisable, mais sa trajectoire coupe la ligne critique : un changement de phase a lieu et l'algorithme requiert beaucoup de temps pour explorer une grande portion du diagramme de phases. Puis il retourne dans la phase satisfaisable et trouve une solution.

La complexité de résolution du problème 3-SAT initial se résume donc essentiellement à celle du problème 2+*p*-SAT critique situé au point de traversée entre les phases. Dans cette région intermédiaire, 3-SAT est exponentiellement difficile à résoudre. L'analyse des trajectoires dans le diagramme de phase permet donc de comprendre et de reproduire quantitativement la complexité de résolution de formules 3-SAT aléatoires. La physique a fourni une analyse de ce qui se passe quand l'algorithme résout une formule SAT.

En 2001, grâce aux notions de variables gelées mises au jour par la physique, nous avons mis au point un nouvel algorithme, de type Davis, Putnam et Loveland, nommé *cnfs*, qui est l'aboutissement pratique des études physiques du problème SAT. Cet algorithme utilise une heuristique qui recherche dans la formule SAT à résoudre des variables gelées, à qui l'on peut attribuer rapidement une valeur *Vrai* ou *Faux*. Cette heuristique détermine de quelle façon une variable est contrainte. Prenons un exemple avec une formule 2-SAT dont la première clause est *x*v*y*. Quand la variable *y* a pour valeur *Faux*, elle contraint la variable *x* à être vraie afin que la clause soit satisfaite. Lorsque la variable *y* apparaît une seule fois dans la formule SAT, la variable *x* est peu contrainte. En revanche, quand la variable *y* est fréquente dans la formule et, de surcroît, sous sa forme négative ¬*y* (il est probable que sa valeur sera *Faux*), elle contraindra la variable *x* à être vraie. L'heuristique examine de la même façon toutes les variables et, au final, identifie les variables le plus vraisemblablement contraintes, donc gelées, et quelle valeur leur donner. Doté d'une telle heuristique, l'algorithme *cnfs* a réduit par un facteur supérieur à trois le temps de résolution par rapport aux meilleurs algorithmes mis au point depuis une dizaine d'années. De plus, il a résolu des formules pour lesquelles certaines études récentes avaient conclu à l'impossibilité de les résoudre par un algorithme quelconque de type Davis, Putnam et Loveland. Pourtant, l'algorithme *cnfs* n'en est qu'à ses débuts.

Les progrès accomplis sur les phénomènes de transition de phase pour les problèmes d'optimisation combinatoire sont exemplaires de l'interaction de deux disciplines. Un exemple éclatant du bénéfice tiré de cette interaction pour le problème SAT est le concept de variables gelées grâce auquel on a pu résoudre des problèmes que l'on croyait très difficilement solubles. La clef des problèmes NP-complets est peut-être au bout de ce type d'interaction.

Simona COCCO est physicienne au Laboratoire de dynamique des fluides complexes, à Strasbourg. Olivier DUBOIS est informaticien au Laboratoire d'informatique de Paris 6. Jacques MANDLER est post-doctorant au Laboratoire d'informatique de Paris 6. Rémi MONASSON est physicien au Laboratoire de Physique théorique de l'École normale supérieure de Paris.