

Répartir sans regret

Mariages stables, flots et coût minimal, de Gale et Shapley à Parcoursup

acceptation différée

paire bloquante

chemin augmentant

coût minimal

Une affectation que personne ne peut contester n'est pas forcément celle qui place le plus d'élèves, ni celle qui les contente le mieux. Ce fascicule raconte pourquoi, et qui l'a découvert.

Un cadeau de fin de cours, à lire par curiosité : rien n'est à rendre. Le projet « Répartir une promotion » reste décrit par sa fiche sur le site et par le README de son squelette ; ce fascicule raconte les idées qui l'entourent et ce qu'elles deviennent au-delà du cours.

1 Pourquoi ce fascicule

Le projet « Répartir une promotion » vous fait écrire trois algorithmes sur les mêmes données : des élèves qui classent des groupes, des groupes qui classent des élèves, et des capacités. L'acceptation différée promet qu'aucun couple élève-groupe ne pourra contester le résultat ; le flot maximal promet de placer le plus d'élèves possible ; le flot à coût minimal promet, à effectif fixé, la plus petite somme de rangs. Vous avez constaté, ou vous constaterez, que ces trois promesses ne désignent pas la même affectation.

Ce fascicule raconte d'où viennent ces idées et ce qu'elles sont devenues. Un article de dix pages paru en 1962 a fini par organiser l'affectation des internes en médecine aux États-Unis et celle des lycéens de New York, et la procédure Parcoursup s'en inspire ; un rapport militaire de 1955 sur un réseau ferré est à l'origine des chemins augmentants que vous avez complétés. Chaque idée vient avec une définition, un exemple déroulé à la main, une esquisse de preuve et une note d'histoire ; nous comparerons ensuite les objectifs.

Il ne contient aucune solution du projet : ni le pas d'acceptation différée, ni la construction du réseau, ni l'augmentation d'un chemin. Les exemples sont nouveaux, et les bouts de code Elixir vérifient des affectations données sans les calculer. Rien n'est à rendre.

2 Une affectation que personne ne conteste

Commençons par la promesse la plus surprenante des trois : qu'aucun élève et aucun groupe ne puissent, ensemble, préférer s'arranger entre eux plutôt que d'accepter le résultat. Nous la définissons, nous la vérifions sur des maps, puis nous suivons tour par tour l'algorithme qui la tient.

2.1 Le problème de Gale et Shapley

En 1962, David Gale et Lloyd Shapley publient dans l'*American Mathematical Monthly* un article de dix pages, « College Admissions and the Stability of Marriage ». Il part de l'admission dans les universités américaines, où des candidats classent des établissements qui les classent en retour et disposent d'un nombre de places. Pour isoler l'idée, les auteurs se ramènent au cas d'une place par établissement et d'autant de candidats que de places ; c'est le problème des *mariages stables*, qui a donné son

nom à toute la théorie. Ils reviennent ensuite aux admissions avec capacités, et c'est exactement notre projet.

Une *affectation* associe à chaque élève au plus un groupe, sans dépasser les capacités. Une paire élève–groupe est *bloquante* quand les deux préféreraient s'arranger entre eux : l'élève accepte ce groupe et le préfère à son affectation actuelle, ou n'a pas d'affectation ; et le groupe a une place libre, ou donne priorité à cet élève sur l'un de ses occupants. Une affectation est *stable* lorsqu'elle n'admet aucune paire bloquante. En clair, personne ne peut contester le résultat en désignant un partenaire qui serait d'accord avec lui.

Vérifions cette définition sur trois élèves et trois groupes d'une place. Ada demande A, B puis C ; Bob demande B, A puis C ; Chloe demande A, B puis C. Le groupe A préfère Bob, puis Ada, puis Chloe ; B et C préfèrent Ada, puis Bob, puis Chloe. Le bout de code ci-dessous cherche les paires bloquantes d'une affectation donnée, avec les maps et la compréhension `for` d'AP4 ; `rang` renvoie la position d'un élément dans une liste, et `occupant` retourne la map pour savoir qui occupe chaque groupe.

```

1  voeux = %{"Ada" => ~w(A B C), "Bob" => ~w(B A C),
2          "Chloe" => ~w(A B C)}
3  priorites = %{"A" => ~w(Bob Ada Chloe), "B" => ~w(Ada Bob Chloe),
4              "C" => ~w(Ada Bob Chloe)}
5  rang = fn liste, x -> Enum.find_index(liste, &(&1 == x)) end
6
7  bloquantes = fn affectation ->
8      occupant = Map.new(affectation, fn {e, g} -> {g, e} end)
9      for {eleve, groupe} <- affectation,
10         autre <- voeux[eleve],
11         rang.(voeux[eleve], autre) < rang.(voeux[eleve], groupe),
12         rang.(priorites[autre], eleve) <
13         rang.(priorites[autre], occupant[autre]),
14         do: {eleve, autre}
15  end
16
17  {bloquantes.(%{"Ada" => "A", "Bob" => "B", "Chloe" => "C"}),
18   bloquantes.(%{"Ada" => "C", "Bob" => "B", "Chloe" => "A"})}
```

```
VALEUR  {}, [{"Ada", "A"}, {"Ada", "B"}]}
```

La première affectation est stable. La seconde donne à Ada son dernier vœu alors que A et B la préfèrent tous deux à leur occupant : elle peut contester auprès de l'un comme de l'autre. Ce bout de code suppose tout le monde placé et une place

par groupe ; le projet traite en plus les places libres, les élèves sans affectation et les capacités.

PIÈGE

La stabilité ne dit rien de la somme des rangs, ni du nombre d'élèves placés. Elle dit seulement qu'aucune contestation à deux n'est possible. Nous verrons aux sections 5 et 6 qu'une affectation stable peut laisser sans place un élève qu'une autre affectation aurait placé, et coûter plus cher qu'une autre.

2.2 Admettre tout de suite, ou garder en attente

L'algorithme fourni par le squelette, qui admet définitivement par tours de vœux, porte un nom dans la littérature : le *mécanisme de Boston*, ainsi baptisé par Atila Abdulkadiroğlu et Tayfun Sönmez (2003) parce que les écoles publiques de Boston l'utilisaient. Son défaut tient en une phrase : un élève qui ose demander d'abord un groupe très demandé et qui n'y entre pas trouve, au tour suivant, les places de son deuxième vœu déjà prises. D'où une paire bloquante dès qu'un groupe le préférerait à son occupant, et d'où une incitation à mettre en tête un vœu « prudent ».

L'*acceptation différée* corrige ce défaut en rendant chaque admission provisoire jusqu'à la fin. Présentons-la par tours simultanés, comme Gale et Shapley : à chaque tour, chaque élève sans place qui a encore des vœux propose au premier groupe qu'il n'a pas encore essayé ; chaque groupe compare les nouveaux candidats à ceux qu'il gardait, conserve les meilleurs selon sa priorité, dans la limite de sa capacité, et rejette les autres, anciens compris ; un élève rejeté reprend sa liste au vœu suivant. On s'arrête quand plus personne ne propose. Le squelette traite une proposition à la fois ; le résultat est le même, car il ne dépend pas de l'ordre des propositions (D. G. McVitie et L. B. Wilson, 1971).

2.3 Quatre élèves, trois groupes, neuf propositions

Déroulons l'algorithme sur un exemple avec une capacité de deux, des priorités différentes par groupe et une cascade de rejets. Les données sont fictives et n'ont rien à voir avec celles du squelette.

élève	vœux	groupe	places	priorité
Ada	C, B, A	A	2	Bob, Ada, Dan, Chloe
Bob	B, C, A	B	1	Chloe, Ada, Bob, Dan
Chloe	C, A, B	C	1	Bob, Chloe, Dan, Ada
Dan	B, C, A			

Au premier tour, chacun propose à son premier vœu : Ada et Chloe à C, Bob et Dan à B. C reçoit deux candidatures pour une place et préfère Chloe à Ada ; B préfère Bob à Dan. Ada et Dan sont rejetés. Au deuxième tour, Ada propose à B, qui la préfère à Bob : Bob, pourtant gardé au tour précédent, est éjecté. Dan propose à C, qui garde Chloe. Au troisième tour, Bob propose à C, qui le préfère à Chloe : Chloe est éjectée à son tour. Dan propose à A, qui a deux places libres et le garde. Au quatrième tour, Chloe propose à A, qui la garde aussi. Plus personne ne propose.

	A · 2 places	B · 1 place	C · 1 place
tour 1	—	Bob, ~~Dan~~	Chloe, ~~Ada~~
tour 2	—	Ada, ~~Bob~~	Chloe, ~~Dan~~
tour 3	Dan	Ada	Bob, ~~Chloe~~
tour 4	Dan, Chloe	Ada	Bob

barré : rejeté à ce tour · rouge : un rejet à lieu · vert : une entrée sans rejet

Chaque case liste les candidatures que le groupe examine au tour indiqué, gardées puis barrées. Bob et Chloe sont éjectés après avoir été gardés : c'est le sens de « différée ». Dan, rejeté deux fois, n'est jamais revenu à son premier vœu.

Le résultat est Ada dans B, Bob dans C, Chloe et Dan dans A, après neuf propositions. Est-il stable ? Ada préférerait C, mais C garde Bob, qu'il préfère ; Bob préférerait B, mais B préfère Ada ; Chloe préférerait C, qui préfère Bob ; Dan préférerait B puis C, qui préfèrent chacun leur occupant. Aucune paire bloquante, et ce n'est pas un hasard : c'est l'objet de la section suivante.

À RETENIR

Une admission définitive crée des paires bloquantes ; une admission provisoire, que seule une meilleure candidature peut défaire, n'en crée aucune. L'élève rejeté reprend sa liste là où il l'avait laissée, jamais au début : un groupe qui l'a rejeté une fois le rejetterait encore.

Vous reconnaissez la forme du code : l'état est une map, file des proposants, vœux restants, admis provisoires et trace ; chaque pas en rend un nouveau, et la boucle fournie le rappelle tant que la file n'est pas vide, comme la calculatrice à pile d'AP1

transmettait sa pile et comme un processus d'AP6 transmet son état. Ce que le cours ne disait pas, c'est pourquoi cette boucle termine et ce que son résultat promet.

3 Ce que l'acceptation différée promet

L'algorithme termine, son résultat est stable, et il est le meilleur résultat stable possible pour ceux qui proposent, qui n'ont en plus aucun intérêt à mentir. Chaque propriété tient en une esquisse de preuve courte ; la dernière a une face cachée, que nous montrerons sur l'exemple à trois élèves.

3.1 Elle termine, et vite

Chaque proposition consomme un vœu, et un élève ne propose jamais deux fois au même groupe. Avec n élèves et m groupes, il y a donc au plus nm propositions, et moins à l'usage, puisque l'on s'arrête dès que chacun est placé ou a épuisé sa liste ; notre exemple en a demandé neuf, sur douze possibles. Donald Knuth, dans un petit livre écrit en français en 1976, *Mariages stables*, analyse le cas de listes tirées au hasard : le nombre moyen de propositions est alors de l'ordre de $n \log n$ pour n élèves et n groupes (L. B. Wilson, 1972).

3.2 Le résultat est stable

Tout repose sur un invariant, que vous pouvez relire sur la figure ci-dessus : ce qu'un groupe garde ne fait que s'améliorer à ses yeux, car un candidat gardé n'est remplacé que par un candidat mieux placé dans sa priorité ; et ce qu'un élève propose ne fait que descendre dans sa liste.

POURQUOI LE RÉSULTAT EST STABLE

Supposons une paire bloquante à l'arrivée : l'élève e préfère le groupe g à son affectation, et g a une place libre ou préfère e à l'un de ses occupants. Comme e descend sa liste dans l'ordre, il a proposé à g avant d'obtenir son groupe actuel, et g l'a rejeté ou éjecté. À ce moment-là, g était complet avec des candidats qu'il préférerait tous à e ; par l'invariant, les occupants finaux de g sont encore meilleurs. Donc g n'a ni place libre ni occupant moins prioritaire que e : contradiction.

Cette preuve est celle de Gale et Shapley ; elle dit aussi pourquoi un élève rejeté ne doit pas revenir en arrière dans sa liste.

3.3 Le meilleur résultat stable pour ceux qui proposent

Il peut exister plusieurs affectations stables. Notre exemple à quatre élèves en a exactement deux : celle que nous avons trouvée, et celle qui place Ada dans A, Bob dans C, Chloe dans B et Dan dans A, meilleure pour A et pour B, moins bonne pour Ada et pour Chloe. Ce n'est pas une coïncidence. Disons qu'un groupe est *possible* pour un élève si une affectation stable au moins le lui attribue. Gale et Shapley montrent par récurrence sur les tours que l'acceptation différée ne rejette jamais un élève d'un groupe possible pour lui : chaque élève obtient donc le meilleur de ses groupes possibles. Le résultat est *optimal pour les proposants* ; si les groupes proposaient, chacun obtiendrait ses meilleurs élèves possibles, et les intérêts des deux côtés sont exactement opposés. Alvin Roth a démontré en 1986 une propriété plus inattendue, le *théorème des hôpitaux ruraux* : le même ensemble d'élèves est placé dans toutes les affectations stables. Changer d'affectation stable ne sauvera donc jamais un élève resté sans place.

3.4 Dire la vérité ne coûte rien, d'un seul côté

Puisque l'élève obtient son meilleur groupe possible, a-t-il intérêt à déclarer une liste qui n'est pas la sienne ? Non : Lester Dubins et David Freedman (1981), puis Alvin Roth (1982), ont montré qu'aucun élève, ni même aucun groupe d'élèves agissant de concert, ne peut obtenir mieux en mentant sur ses vœux. C'est le sens du titre de ce fascicule : celui qui propose peut classer sincèrement, sans calcul ni regret. Roth a montré du même coup qu'aucune procédure produisant toujours une affectation stable n'a cette propriété pour les deux côtés à la fois.

L'autre côté peut donc avoir intérêt à manipuler, et David Gale et Marilda Sotomayor (1985) en ont donné l'exemple classique. Reprenons l'instance à trois élèves de la section 1. Avec les vraies priorités, le premier tour place Ada dans A et Bob dans B, et Chloe, rejetée par A puis par B, entre dans C : A obtient Ada, sa deuxième priorité. Supposons maintenant que A déclare la fausse priorité Bob, Chloe, Ada. Au premier tour, A garde Chloe et rejette Ada ; Ada propose à B, qui la préfère à Bob et l'éjecte ; Bob propose à A, qui le préfère à Chloe et l'éjecte ; Chloe est rejetée par B puis entre dans C. Résultat : Ada dans B, Bob dans A, Chloe dans C. En mentant, A a obtenu Bob, sa vraie première priorité ! Cette affectation est d'ailleurs stable au sens des vraies priorités, comme le bout de code de la section 1 vous le confirmera : c'est la meilleure affectation stable pour les groupes.

QUI PROPOSE A L'AVANTAGE

Dans le projet, les priorités sont une donnée annoncée, par ex. un tirage au sort, et personne ne les manipule. Dans un système réel, décider quel côté propose est une décision politique : elle fixe qui obtient la meilleure affectation stable et qui peut classer sans arrière-pensée.

4 Des internes aux lycéens

L'article de 1962 est resté une curiosité mathématique pendant vingt ans, jusqu'à ce que des économistes reconnaissent l'algorithme dans des procédures existantes, puis l'installent dans d'autres. En 2012, le prix de la Banque de Suède en sciences économiques en mémoire d'Alfred Nobel a récompensé Lloyd Shapley et Alvin Roth « pour la théorie des allocations stables et la pratique de la conception de marchés » ; David Gale, mort en 2008, n'a pas pu le partager.

4.1 Les internes américains

Depuis 1952, les jeunes diplômés en médecine des États-Unis et les hôpitaux qui les recrutent passent par une chambre de compensation centrale, le *National Resident Matching Program* (NRMP) : chacun dépose une liste classée, et un algorithme produit l'affectation. En 1984, Roth a montré que cet algorithme était équivalent à l'acceptation différée où les hôpitaux proposent ; les praticiens l'avaient donc inventé dix ans avant Gale et Shapley, sans en connaître les propriétés. Roth et Elliott Peranson l'ont ensuite redessiné pour que les candidats proposent, en y intégrant les couples qui cherchent deux postes dans la même ville ; cette version est en service depuis 1998 (Roth et Peranson, 1999) et affecte chaque année des dizaines de milliers de candidats.

4.2 Le choix d'école

Abdulkadiroğlu et Sönmez (2003) ont proposé de traiter l'affectation des élèves aux écoles publiques comme un problème d'admissions : les familles classent des écoles, dont les priorités viennent de règles publiques, par ex. la fratrie et la proximité, puis d'un tirage au sort. Ils ont montré que le mécanisme de Boston pousse les familles à mentir, et recommandé l'acceptation différée. New York l'a adoptée en 2003 pour ses lycées, avec l'aide d'Abdulkadiroğlu, de Parag Pathak et de Roth, et Boston en 2005.

4.3 Parcoursup

En France, la procédure Admission Post-Bac, utilisée de 2009 à 2017, demandait aux candidats une liste classée de vœux. Parcoursup, qui l’a remplacée en 2018, ne la demande plus. Le code de ses algorithmes et le document qui les présente sont publiés depuis 2018, et ce document décrit la mécanique suivante : chaque formation classe ses candidats, et un *ordre d’appel* est calculé à partir de ce classement en y intégrant les taux légaux, par ex. de boursiers ; chaque jour, chaque formation envoie des propositions d’admission aux candidats suivants dans son ordre d’appel, dans la limite de ses places non encore acceptées ou en attente de réponse ; un candidat qui accepte une proposition conserve ses autres vœux en attente, et peut renoncer plus tard à ce qu’il a accepté au profit d’une proposition qu’il préfère. Le document ne cite pas Gale et Shapley, mais vous reconnaissez la forme : une acceptation différée où les formations proposent, la liste de vœux du candidat étant remplacée par ses réponses au fil des jours. Le prix se lit dans le calendrier : la phase principale de 2018 courait du 22 mai au 5 septembre, là où une acceptation différée sur des listes classées se calcule en quelques secondes ; en échange, un lycéen n’a pas à classer ses vœux avant de savoir lesquels lui sont ouverts.

5 Placer le plus d’élèves possible

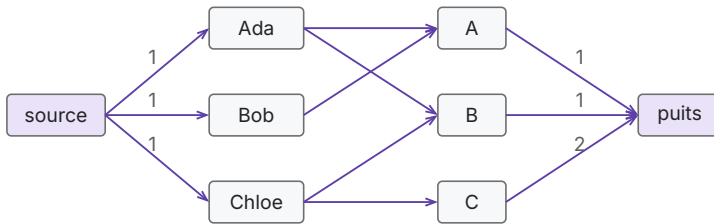
Changeons de question, comme le fait le projet : oublions les priorités et cherchons combien d’élèves peuvent être placés dans leurs premiers vœux. C’est une question de graphe, et sa réponse est venue d’un tout autre problème.

5.1 Un couplage est un flot

Dessignons les élèves à gauche, les groupes à droite, et un arc d’un élève vers chaque groupe qu’il accepte : c’est un *graphe biparti*, c’est-à-dire un graphe dont les sommets se répartissent en deux côtés et dont les arcs vont toujours d’un côté à l’autre. Un *couplage* est un ensemble d’arcs qui n’utilise chaque élève qu’une fois et chaque groupe au plus autant de fois que sa capacité. Placer le plus d’élèves possible, c’est chercher un couplage de taille maximale.

Un *flot* est une quantité qui circule d’une *source* vers un *puits* en respectant la *capacité* de chaque arc, et en se conservant à chaque sommet intermédiaire, qui transmet exactement ce qu’il reçoit. Il suffit d’ajouter une source reliée à chaque élève par un arc de capacité un, et un puits que chaque groupe rejoint par un arc portant sa capacité ; les arcs élève–groupe portent la capacité un. Chaque unité de flot entière trace alors un chemin source, élève, groupe, puits, c’est-à-dire une affectation. Voici

le réseau pour Ada, qui accepte A puis B, Bob, qui n'accepte que A, et Chloe, qui accepte C puis B, avec deux places dans C.



arcs élève–groupe : les vœux, capacité 1 chacun

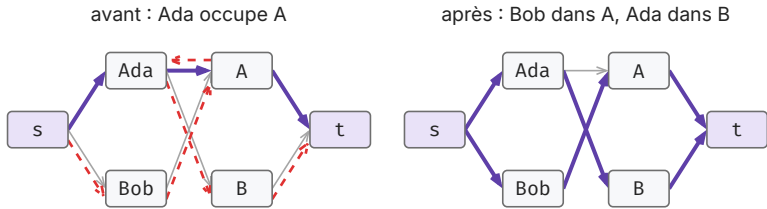
Le réseau de flot du couplage. Les étiquettes sont les capacités : un sur les arcs de la source, les places du groupe sur les arcs vers le puits. Une unité de flot entière suit toujours un chemin source, élève, groupe, puits.

Le problème du flot maximal a été posé et résolu par Lester Ford et Delbert Fulkerson en 1956, à la RAND Corporation ; d'après l'enquête historique d'Alexander Schrijver (2002), leur motivation était un rapport de 1955 de leurs collègues T. E. Harris et F. S. Ross sur la capacité du réseau ferroviaire reliant l'Union soviétique à l'Europe de l'Est, et sur la manière la moins coûteuse de la couper. Les deux questions n'en font qu'une : la valeur du flot maximal est égale à la capacité de la *coupe* minimale, le plus petit ensemble d'arcs qui sépare la source du puits.

5.2 Les chemins augmentants, pas à pas

L'algorithme de Ford et Fulkerson part d'un flot nul et l'augmente d'un chemin à la fois. Le *graphe résiduel* décrit ce qu'il reste possible de changer : chaque arc conserve sa capacité non utilisée, et chaque unité envoyée crée en sens inverse une capacité de retour, qui permet de l'annuler. Un *chemin augmentant* est un chemin de la source au puits dans ce graphe ; l'emprunter envoie une unité de plus, en annulant au passage les affectations dont il prend l'arc de retour.

Suivons l'algorithme sur le réseau de la figure, en explorant les élèves dans l'ordre Ada, Bob, Chloe. Le premier chemin trouvé est source, Ada, A, puits : Ada occupe A. Le deuxième est source, Chloe, C, puits : Chloe occupe C. Pour le troisième, Bob ne peut aller qu'en A, dont l'arc vers le puits est saturé ; mais le graphe résiduel contient l'arc de retour de A vers Ada, puis l'arc d'Ada vers B, puis l'arc de B vers le puits, libre. Le chemin source, Bob, A, Ada, B, puits existe donc, et l'emprunter retire Ada de A, place Bob dans A et Ada dans B.



pointillés : le chemin augmentant ; A vers Ada est l'arc de retour

Chloe est omise pour la lisibilité. Avant : Ada occupe A et Bob est bloqué. Le chemin augmentant passe par l'arc de retour de A vers Ada, qui annule l'affectation d'Ada. Après : trois unités de flot au total, personne n'occupe deux places.

Pourquoi s'arrêter quand aucun chemin augmentant n'existe ? Parce que les sommets encore accessibles depuis la source dans le graphe résiduel définissent alors une coupe dont tous les arcs sortants sont saturés ; sa capacité est égale au flot courant, qui ne peut donc pas être dépassé. Reste à choisir le chemin : Jack Edmonds et Richard Karp ont montré en 1972 que prendre à chaque fois le plus court, celui qui fournit un parcours en largeur, borne le nombre d'augmentations par $O(VE)$ pour V sommets et E arcs, et c'est la raison du parcours en largeur fourni par le squelette. Pour le couplage biparti, John Hopcroft et Richard Karp (1973) augmentent plusieurs chemins courts à la fois et terminent en $O(E\sqrt{V})$, ce qui compte pour des millions d'arcs.

5.3 Un certificat d'impossibilité

Deux théorèmes plus anciens disent la même chose que la coupe, dans le langage des graphes bipartis. Dénes Kőnig a montré en 1931 que la taille du couplage maximal d'un graphe biparti est égale au plus petit nombre de sommets qui touchent tous les arcs ; Philip Hall a donné en 1935 la condition pour que tous les élèves soient placés : il faut et il suffit que tout ensemble d'élèves accepte, au total, au moins autant de places qu'il compte d'élèves. Un ensemble qui la viole est un certificat d'impossibilité lisible par tous. Le bout de code ci-dessous compte les places accessibles à un ensemble d'élèves, en réunissant avec `Enum.flat_map/2` et `Enum.uniq/1` les groupes acceptés.

```

1  voeux = %{"Ada" => ~w(A), "Bob" => ~w(A B), "Chloe" => ~w(A)}
2  capacites = %{"A" => 1, "B" => 1}
3  groupe_demandeurs = ~w(Ada Chloe)
4
5  places =
6    groupe_demandeurs
7    |> Enum.flat_map(&voeux[&1])
8    |> Enum.uniq()
9    |> Enum.map(&capacites[&1])
10   |> Enum.sum()
11
12 {length(groupe_demandeurs), places}

```

VALEUR {2, 1}

Deux élèves n'acceptent qu'une seule place : l'un des deux restera sans affectation, quel que soit l'algorithme. Quand le flot maximal s'arrête, les élèves encore accessibles depuis la source forment un tel ensemble.

5.4 Stabilité et effectif ne vont pas ensemble

Revenons aux priorités, et donnons à A la priorité Ada puis Bob. L'acceptation différée place Ada dans A et Chloe dans C ; Bob, rejeté par A, n'a plus de vœu et reste sans place. Le flot maximal, lui, place les trois, Bob dans A, Ada dans B et Chloe dans C, mais Ada préfère A à B et A la préfère à Bob : la paire Ada-A conteste. Par le théorème des hôpitaux ruraux, toute affectation stable laisse Bob dehors.

À RETENIR

Le flot ne consulte pas les priorités : il compte des places. L'acceptation différée respecte les priorités : elle peut laisser des places vides. Les deux promesses sont exactes, et elles ne parlent pas de la même chose.

6 Le meilleur coût, et à quel prix

Parmi les affectations qui placent le plus d'élèves possible, laquelle les contente le mieux ? Le projet additionne les rangs des vœux obtenus et cherche la plus petite somme. Ce coût se calcule avec un `Enum.reduce/3` à la manière du chapitre 4 ; voici

celui de l'affectation stable de la section 1, et celui d'une autre affectation que nous retrouverons plus bas :

```

1  voeux = %{"Ada" => ~w(C B A), "Bob" => ~w(B C A),
2          "Chloe" => ~w(C A B), "Dan" => ~w(B C A)}
3
4  cout = fn affectation ->
5    Enum.reduce(affectation, 0, fn {eleve, groupe}, acc ->
6      acc + Enum.find_index(voeux[eleve], &(&1 == groupe))
7    end)
8  end
9
10 stable = %{"Ada" => "B", "Bob" => "C", "Chloe" => "A", "Dan" => "A"}
11 minimale = %{"Ada" => "C", "Bob" => "B",
12             "Chloe" => "A", "Dan" => "A"}
13 {cout.(stable), cout.(minimale)}
```

VALEUR {5, 3}

Cette question est plus ancienne que les flots, et elle a sa propre histoire.

6.1 La méthode hongroise

Le *problème d'affectation* classique donne n personnes, n tâches et un coût pour chaque paire, et cherche l'appariement de coût total minimal. Harold Kuhn en a publié en 1955 un algorithme qu'il a appelé *méthode hongroise*, en hommage à König et à Jenő Egerváry, dont les théorèmes de 1931 en fournissent le cœur : on ajuste des potentiels sur les lignes et les colonnes de la matrice des coûts jusqu'à ce qu'un couplage parfait n'utilise que des arcs de coût réduit nul. James Munkres a montré en 1957 que la méthode termine en un nombre d'opérations polynomial, et les bibliothèques d'optimisation l'implémentent encore sous ce nom.

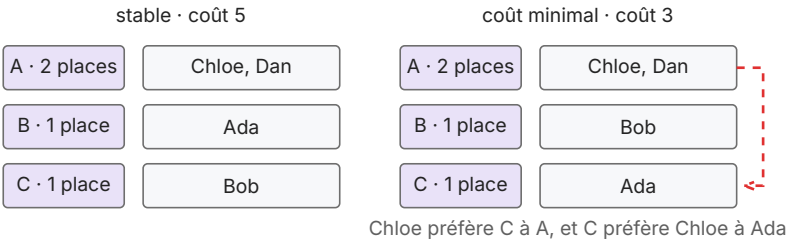
6.2 Le flot à coût minimal

Notre problème a des capacités et des élèves parfois non placés ; le cadre naturel est le *flot à coût minimal*, où chaque arc porte un coût par unité et où l'on cherche, pour une valeur de flot donnée, le flot le moins cher. La méthode des *plus courts chemins successifs*, formalisée par Robert Busacker et Paul Gowen en 1960, reprend Ford et Fulkerson en choisissant à chaque étape le chemin augmentant le moins cher. Un arc de retour y porte le coût opposé de son arc aller, puisque reprendre une affectation rembourse son coût ; des coûts négatifs apparaissent donc dès la première augmentation, et c'est pourquoi le squelette fournit l'algorithme de Bellman et Ford (Ford, 1956 ; Richard Bellman, 1958), qui les accepte, là où celui d'Edsger Dijkstra

(1959) les refuse. L’invariant est que, après chaque augmentation, le flot courant est le moins cher de sa valeur, d’où le coût minimal à effectif maximal quand les chemins s’épuisent.

6.3 Trois objectifs, trois affectations

L’affectation stable de la section 1 coûte cinq. Or la moins chère des douze affectations complètes de cet exemple coûte trois : Ada dans C, Bob dans B, Chloe et Dan dans A. Elle donne à Ada et à Bob leur premier vœu au lieu du deuxième et laisse Chloe et Dan où ils étaient ; aucun élève n’y perd. Mais Chloe préfère C à A, et C préfère Chloe à Ada : la paire Chloe–C conteste, et Dan–C aussi.



Le même exemple, deux affectations. Celle de droite coûte moins et ne fait reculer personne, mais C la conteste avec Chloe. Le flot à coût minimal ne consulte pas les priorités des groupes : sa promesse porte sur la somme des rangs des élèves.

Cet exemple illustre un fait général, signalé par Roth dès 1982 : la meilleure affectation stable pour les élèves peut être dominée, pour eux, par une affectation instable. La stabilité protège les priorités des groupes, et cette protection a un prix. Voici les trois promesses :

objectif	ce qu’il garantit	ce qu’il ignore
stabilité	aucune contestation à deux ; respect des priorités	le nombre de placés, la somme des rangs
effectif maximal	le plus grand nombre d’élèves placés	les priorités, les rangs obtenus
coût minimal	la plus petite somme de rangs à effectif fixé	les priorités ; qui paie pour la somme

La somme des rangs est un objectif *utilitariste* : elle accepte qu’un élève reçoive son dernier vœu si deux autres y gagnent un rang, là où un objectif d’*équité* regarderait le pire rang ; et comme une affectation vide a un coût nul, la comparer n’a de sens qu’à effectif égal. Aucun algorithme ne maximise tout à la fois, parce que ces objectifs

désignent des affectations différentes sur le même exemple ; le choix entre eux n'est pas algorithmique.

7 Au-delà du projet

Le même modèle sert bien au-delà des groupes de TP. Les *échanges de reins* en sont l'application la plus frappante : un patient dont le donneur vivant est incompatible avec lui peut échanger ce donneur avec un autre couple dans la même situation. Roth, Sönmez et Utku Ünver ont proposé en 2004 de les organiser à grande échelle, par des cycles et des chaînes de couples, à partir des cycles d'échange de Shapley et Herbert Scarf (1974) ; de tels programmes existent aujourd'hui dans plusieurs pays, dont la France. Les *enchères publicitaires* des moteurs de recherche sont, elles, une affectation entre annonceurs et emplacements, dont Benjamin Edelman, Michael Ostrovsky et Michael Schwarz (2007) ont relié l'équilibre aux affectations stables.

Que change-t-on en touchant aux règles ? Les *capacités* ne changent rien, à une hypothèse près : la préférence d'un groupe pour un ensemble d'élèves doit découler de sa priorité sur les individus, ce que fait la sélection fournie par le squelette. Les *quotas* qui réservent des places, par ex. le taux de boursiers de Parcoursup, se traitent en adaptant l'ordre d'appel. En revanche, des *minima* par groupe peuvent rendre la stabilité impossible, et même difficile à décider (Péter Biró, Tamás Fleiner, Robert Irving et David Manlove, 2010) ; c'est pourquoi le projet les exclut. Les *égalités* dans les priorités sont le cas réel des écoles ; un tirage au sort les départage, mais Aytekin Erdil et Haluk Ergin (2008) ont montré qu'il peut alors rester des échanges qui amélioreraient des élèves sans en léser aucun. Les *couples*, enfin, qui veulent deux postes au même endroit, peuvent faire disparaître toute affectation stable (Roth, 1984).

Pour aller plus loin

College Admissions and the Stability of Marriage, David Gale et Lloyd S. Shapley (1962).

<https://iqua.ece.toronto.edu/baochun/ece1771f/papers/Gale-Shapley-1962.pdf>

Dix pages lisibles sans autre bagage que ce fascicule : l'algorithme, la preuve de stabilité et l'optimalité y sont déjà.

Mariages stables et leurs relations avec d'autres problèmes combinatoires, Donald E. Knuth (1976).

<https://www-cs-faculty.stanford.edu/~knuth/ms.html>

Six conférences données à Montréal, en français : le treillis des affectations stables et l'analyse en moyenne.

Deferred Acceptance Algorithms: History, Theory, Practice, and Open Questions, Alvin E. Roth (2008).

<https://www.nber.org/papers/w13225>

L'histoire du NRMP et des écoles de New York et de Boston, par celui qui l'a faite.

Stable Allocations and the Practice of Market Design, comité du prix de sciences économiques, Académie royale des sciences de Suède (2012).

<https://www.nobelprize.org/uploads/2018/06/advanced-economicsciences2012.pdf>

Le document scientifique du prix 2012, de la théorie aux applications, sur des exemples de la taille des nôtres.

Maximal Flow Through a Network, L. R. Ford Jr. et D. R. Fulkerson (1956).

<https://doi.org/10.4153/CJM-1956-045-5>

Cinq pages en accès libre : le flot, les chemins augmentants et le théorème flot maximal, coupe minimale.

Algorithmes de Parcoursup : code et documentation, ministère de l'Enseignement supérieur et de la Recherche (2018).

<https://gitlab.mim-libre.fr/parcoursup/algorithmes-de-parcoursup>

Le code Java publié et un document de présentation par année dans `doc/` : cherchez-y l'ordre d'appel, puis l'acceptation différée.

Une question pour la route

Dans Parcoursup, ce sont les formations qui proposent, et le candidat ne classe pas ses vœux : il répond, jour après jour, à ce qu'on lui envoie. D'après la section 3, de quel côté se trouve l'avantage stratégique, et que pourrait faire un candidat, ou une formation, pour en tirer parti ? Puis imaginez la procédure inverse, où les candidats classent leurs vœux et où l'acceptation différée tourne en une nuit : qui y gagnerait, qui y perdrait, et qu'achètent les semaines d'attente ? Il n'y a pas de corrigé ; pour en discuter, vous pouvez me contacter par email, comme d'habitude.