

Algorithmique et programmation fonctionnelle

UE INF 201,231,251

PROJET – COMMENT VOTER ?

Pierre Wulles, Nicolas Basset

2025

Consignes :

- Le langage de programmation utilisé sera OCaml
- Vous pouvez (et devez quand c'est nécessaire !) commenter votre code en le mettant entre `(*...*)`
- Les programmes devront être écrits en utilisant la partie d'OCaml enseignée dans cette UE, c'est à dire en particulier sans boucles `for` `while`, ni référence et tableaux.
- Dans toute question vous pouvez utiliser une fonction d'une question précédente et ce même si vous n'avez pas su écrire la fonction
- Vous ne pouvez utiliser une fonction auxiliaire que si vous la définissez et la réalisez vous même avant

Le but de ce sujet est de comparer plusieurs méthodes de scrutins. Expliquer les avantages et inconvénients de chacun.

1 Introduction

La majorité des démocraties dans le monde a recours au scrutin uninominal à un ou deux tours pour élire ses représentants. Or ce mode de scrutin peut poser problèmes et paradoxes, voici quelques exemples :

- **Présidentielles de 1988** : François Mitterand gagne contre Jacques Chirac mais aurait perdu contre Raymond Barre qui n'arrive pas au second tour.
- **Présidentielles de 1995** : Jacques Chirac gagne contre Lionel Jospin au second tour, mais si Philippe de Villiers n'était pas candidat, Edouard Balladur aurait pu devancer Chirac au premier tour.
- **Présidentielles US de 2000** : G.W. Bush est élu mais si Ralph Nader ne s'était pas présenté, Al Gore aurait été élu.
- **Présidentielles de 2002** : Lionel Jospin était donné gagnant contre Jean-Marie Le Pen et contre Jacques Chirac au second tour mais c'est Jacques Chirac qui a été élu face à Jean-Marie Le Pen. Vraisemblablement, sans les candidatures de Christiane Taubira et Jacques Chevènement, Lionel Jospin aurait été au second tour. Jacques Chirac avait lui-même soutenu la candidature de Christiane Taubira pourtant dans le camp opposé. Charles Pasqua avait renoncé à se présenter au dernier moment de peur de priver la droite d'un second tour.
- **Présidentielles de 2007** : Nicolas Sarkozy est élu mais selon tous les sondages François Bayrou était donné gagnant contre tous les candidats au second tour.

Ces exemples le montrent, le modèle traditionnel ne semble pas traduire la réelle volonté du peuple car le gagnant dépend en partie du jeu des candidatures. En effet l'apparition de « petits candidats » peut faire perdre un camp par rapport à un autre et la stratégie dite de « vote utile » peut pousser les électeurs à voter pour un candidat dont ils ne veulent pas réellement en pensant qu'il a plus de chance qu'un autre. Le vote blanc n'est pas non plus reconnu par ces systèmes. Le but de ce sujet est d'explorer d'autres méthodes de scrutin qui ont pour objectif de résoudre les problèmes précédemment mentionnés.

2 Scrutin uninominal

Attention

Dans cette partie toutes les fonctions utilisées doivent être spécifiés et réalisés. En particulier on n'utilisera aucune fonction issue des modules de la librairie standard d'OCaml.

Un bulletin sera représenté par une chaîne de caractères (qui représente un candidat) et une urne sera représentée par une liste de bulletin. Le score d'un candidat est le nombre de votants pour ce candidat. Le panel est la liste des candidats. Un dépouillement est une liste dont les éléments sont des couples `candidat*score`. On supposera que l'on a jamais d'ex-aequo entre deux candidats car en pratique cela est très improbable.

Question 1 Proposer des types OCaml pour représenter un candidat, un bulletin, une urne, le score d'un candidat, un panel de candidats et un dépouillement.

On se place pour le moment dans le cas d'un scrutin uninominal à un tour.

Pour l'exemple considérons que l'on a trois candidats : Kenny, Kyle et Stan et 16 électeurs. L'urne, le panel et le dépouillement ressemblent à ceci :

```
let lc1 = ["Eric"; "Kyle"; "Stan"];;  
let u1 = ["Eric"; "Kyle"; "Stan"; "Kyle"; "Kyle"; "Stan"; "Eric"; "Eric"; "Kyle"; "Eric"; "Stan"; "Eric";  
         "Eric"; "Eric"; "Stan"; "Stan"];;  
let dep1 = [("Eric", 7); ("Kyle", 4); ("Stan", 5)];;
```

Question 2 Écrire une fonction OCaml `compte c u` qui pour une urne donnée renvoie le score du candidat `c`. Par exemple :

```
val compte : candidat -> urne -> score = <fun>  
# compte "Eric" u1;;  
- : score = 7
```

Question 3 Écrire une fonction `depouiller lc u` qui en prenant le panel et l'urne renvoie le dépouillement correspondant. Par exemple : **Tous les candidats** du panel doivent apparaître dans le résultat, même si ils n'obtiennent aucune voix, **l'ordre** dans lequel ils apparaissent doit **être celui du panel**.

```
val depouiller : panel -> urne -> depouillement = <fun>  
# depouiller lc1 u1;;  
- : depouillement = [("Eric", 7); ("Kyle", 4); ("Stan", 5)]
```

Question 4 Écrire une fonction `union r1 r2` qui fait l'union des dépouillement. Cela pourra par exemple représenter la mise en commun des résultats de deux bureaux de votes.

```
val union : depouillement -> depouillement -> depouillement = <fun>
```

Question 5 Écrire une fonction `max_depouille l` qui en prenant le dépouillement renvoie le candidat qui a le meilleur score, et son score.

```
val max_depouiller : depouillement -> candidat * score = <fun>
```

Question 6 En utilisant les fonctions précédentes, écrire une fonction `vainqueur_scrutin_uninominal u lc` qui pour une urne et un panel donnés renvoie le vainqueur de l'élection, c'est à dire le candidat ayant obtenu le plus de voix. On supposera qu'il n'y a pas d'égalité possible.

```
val vainqueur_scrutin_uninominal : urne -> panel -> candidat = <fun>
```

Question 7 En pratique pour donner plus de poids au résultat d'un scrutin uninominal on veut s'assurer que le candidat élu recueille au moins la moitié des suffrages. Pour cela on organise un **second tour** avec les deux premiers candidats du premier tour. C'est ce système qui est utilisé en France pour les élections présidentielles ou législatives. Écrire une fonction `deux_premiers` qui à partir du dépouillage d'une urne extrait les deux premiers.

Question 8 Dans cette question, nous allons au travers d'une étude de cas, mettre en évidence un problème de ce mode de scrutin. On suppose maintenant que, toutes choses égales par ailleurs, un quatrième candidat fait son apparition, en conséquence certains électeurs reportent leurs votes sur ce candidat :

```
let lc2 = ["Eric"; "Kyle"; "Stan"; "Kenny"];;  
let u2 = ["Kenny"; "Kyle"; "Kenny"; "Kyle"; "Kyle"; "Kenny"; "Eric"; "Eric"; "Kyle"; "Eric"; "Stan"; "Eric";  
         "Eric"; "Eric"; "Stan"; "Stan"];;
```

Quel est le problème? Ce phénomène se nomme **paradoxe d'Arrow**. Identifier les situations similaires dans l'introduction et trouver un exemple de paradoxe d'Arrow dans le paysage politique français récent, par exemple dans l'élection présidentielle de 2022.

Question 9 Identifier précisément la source du problème dans le scrutin **uninominal**.



FIGURE 1 – Philippe Geluck



FIGURE 2 – Stan, Kyle, Eric et Kenny (Southpark)

3 Jugement majoritaire

Attention

Dans cette partie il n'est plus possible d'écrire des fonctions récursives (sauf pour `vainqueur_jm_triee`), vous devrez utiliser les fonctions du module `OCaml List` comme par exemple `List.map`, `List.hd`, `List.tl`, `List.nth`, `List.assoc` ...

Le jugement majoritaire est une méthode mise au point en 2007 par deux chercheurs du CNRS : Michel Balinski et Rida Laraki. L'idée est que chaque électeur attribue aux candidats des mentions, la détermination du gagnant se fait par la **médiane** plutôt que par la **moyenne**. Le jugement majoritaire a par exemple été utilisé lors de la primaire populaire de 2022.

Voici un exemple de bulletin pour une élection à 4 candidats :

	Eric	Kyle	Stan	Keny
TB (Très bien)	×			
B (Bien)				
AB (Assez bien)		×		
P (Passable)				×
I (Insuffisant)				
R (à Rejeter)			×	

Question 10 Dans le cadre d'une élection avec 12 candidats comme celle de 2022, calculer le nombre de bulletin différent qu'un électeur peut mettre dans l'urne. Comparer aux 13 possibilités (12 candidats + nul/blanc) du scrutin uninominal. Commenter.

Question 11 Proposer un type énuméré `OCaml mention` pour représenter une mention, puis un type `bulletin_jm` pour représenter un bulletin de jugement majoritaire comme une liste de couples (candidat, mention). Proposer un type `urne_jm` comme une liste de `bulletin_jm`. Enfin, proposer un type `depouillement_jm` comme une liste de couples (candidat, mention list) qui donne pour chaque candidat la liste des mentions obtenues. Par exemple le bulletin précédent est

```
val b : bulletin_jm = ["Eric",TB; "Kyle",AB; "Stan",R; "Kenny",P]
```

Question 12 (Difficile) On va maintenant lister toutes les mentions obtenues par chaque candidat dans une urne, pour cela on va écrire une fonction `depouille_jm` qui associe à chaque candidats l'ensemble des mentions qu'il obtient à partir des bulletins présents dans l'urne. Par exemple la liste :

```
val u : urne_jm = [ ["Eric",TB; "Kyle",AB; "Stan",R; "Kenny",P]; (* Premier bulletin *)
  ["Eric",AB; "Kyle",AB; "Stan",R; "Kenny",TB]; (* Second bulletin *)
  ["Eric",TB; "Kyle",R; "Stan",R; "Kenny",TB]] (* Troisième bulletin *)
```

deviendra :

```
val ms : depouillement_jm = [ ("Eric", [TB; AB; TB]); (* Mentions du premier candidat *)
  ("Kyle", [AB; AB; R]); (* Mentions du second candidat *)
  ("Stan", [R; R; R]); (* Mentions du troisième candidat *)
  ("Kenny", [P; TB; TB])] (* Mentions du quatrième candidat *)
```

Aide

Une manière de faire est la suivante. Pour chaque candidat du panel, on crée un couple avec le nom du candidat et une liste de mention correspondant au candidat. La liste de mention d'un candidat est obtenu en prenant pour chaque bulletin la mention correspondant au candidat dans ce bulletin. On pourra utiliser pour cela la fonction `List.assoc`.

Question 13 Écrire une fonction `tri_mentions` qui trie les mentions des candidats. Par exemple sur la liste précédente on obtiendra le résultat suivant. On pourra utiliser la fonction de tri de liste définie par

`let tri (l:'a list):'a list = List.sort compare l;;` Noter bien que si le type mention a été correctement défini, on a `R < I < ... < TB`. Le dépouillement précédent une fois trié est :

```
val ms_triee : depouillement_jm =
  [ ("Eric", [AB; TB; TB]); (* Mentions du premier candidat *)
    ("Kyle", [R; AB; AB]); (* Mentions du second candidat *)
    ("Stan", [R; R; R]); (* Mentions du troisième candidat *)
    ("Kenny", [P; TB; TB])] (* Mentions du quatrième candidat *)
```

Question 14 Écrire une fonction `mediane` qui renvoie la médiane d'une liste triée, on choisira la convention suivante : la médiane est l'élément $n/2$ avec $/$ qui désigne ici la division entière comme en OCaml et les indices des listes commencent à 0. Par exemple (`mediane [1;2;3]`) alors que (`mediane [1;2;3;4]`) vaut 3.

Question 15 Écrire une fonction `meilleure_mediane` qui renvoie la meilleure mediane à partir d'un dépouillement trié.

Question 16 Écrire une fonction `supprime_perdants` qui prend en entrée un dépouillement trié et une mention m (correspondant à la meilleur médiane) et supprime tous les candidats qui ont une médiane inférieur à m . Par exemple :

```
supprime_perdants ms_triee TB=[("Eric", [AB; TB; TB]); ("Kenny", [P; TB; TB])]
```

Question 17 Mais il peut rester plusieurs candidats avec la même médiane ! On va alors supprimer cette médiane et regarder quelle est la nouvelle médiane. On réitère alors le procédé de la fonction précédente jusqu'à arriver au cas où il ne reste plus qu'un seul candidat (on suppose toujours qu'il n'y a pas d'égalité). Créer une fonction `supprime_meilleure_mediane` qui supprime la médiane de chaque candidat. Par exemple sur la liste précédente le résultat sera :

```
supprime_medianes ms_triee = [("Eric", [AB; TB]); ("Kenny", [P; TB])]
```

Aide

On pourra écrire et utiliser une fonction auxiliaire récursive qui étant donné un indice i et une liste l supprime le i ème élément de l .

Question 18 (Difficile) Écrire une fonction `vainqueur_jm_triee` qui à un dépouillement trié donne le vainqueur s'il existe ou retourne la chaîne de caractères "EGALITE" sinon.

Remarque

Dans le cas d'égalité, les médianes des listes de mentions des candidats ex-aequo sont successivement supprimées jusqu'à ce que les listes soient vides.

Question 19 Pourquoi la fonction `vainqueur_jm_triee` termine t'elle ?

Question 20 Finalement, écrire une fonction `trouve_vainqueur_jm` qui, à partir d'une urne, détermine la vainqueur d'un scrutin au jugement majoritaire s'il existe ou retourne la chaîne de caractères "EGALITE" sinon.

Question 21 Expliquer succinctement comment le jugement majoritaire résout le problème du vote utile et trouver au moins une critique que l'on peut faire au jugement majoritaire.

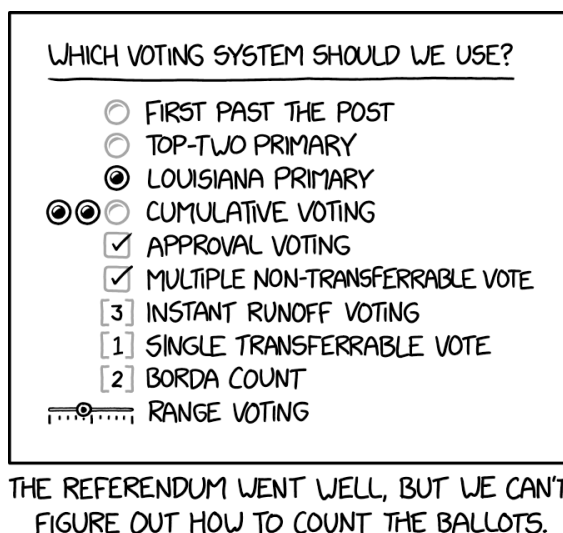


FIGURE 3 – xkcd.com

4 Recomptons les voix

Après avoir exploré les différents système de vote qu'il est possible de mettre en place, nous allons recompter les voix du premier tour de l'élection présidentielle de 2022. On se limitera aux résultats de la région Auvergne-Rhône-Alpes et on se basera sur les données rendues accessibles par le ministère de l'intérieur ¹.

Pour hiérarchiser les données nous allons les représenter sous forme d'arbre représentant des zones de plus en plus petites. La racine de l'arbre sera la région (ici Auvergne-Rhône-Alpes) puis les branches seront les départements et enfin les feuilles seront les bureaux de votes que l'on assimilera ici aux communes pour simplifier (même si en réalité on peut avoir plusieurs bureaux de vote dans une seule commune...) Il faudra donc introduire un type `ville` pour nommer les bureaux de vote et un type `région` qui représentera soit une région soit un département. On donne ci dessous un exemple d'un arbre créé dans le cas (fictif) ou la région Auvergne-Rhône-Alpes ne contiendrait que deux départements, l'Isère et la Drome, ne contenant que quelques villes (cet arbre est trouvable dans le fichier `etd.ml`) :

1. data.gouv.fr

```

val ara : arbre =
N (Reg "Auvergne-Rhône-Alpes",
  [N (Dpt "Drôme",
    [Bv ("Valence",
      [("ARTHAUD", 161); ("ROUSSEL", 595); ("MACRON", 7756); ("LASSALLE", 590);
      ("LE PEN", 4679); ("ZEMMOUR", 2080); ("MÉLENCHON", 8398);
      ("HIDALGO", 519); ("JADOT", 1701); ("PÉCRESSE", 1423); ("POUTOU", 186);
      ("DUPONT-AIGNAN", 573)]]);
    Bv ("Romans-sur-Isère",
      [("ARTHAUD", 181); ("ROUSSEL", 371); ("MACRON", 4030); ("LASSALLE", 334);
      ("LE PEN", 3270); ("ZEMMOUR", 1072); ("MÉLENCHON", 4108);
      ("HIDALGO", 251); ("JADOT", 850); ("PÉCRESSE", 631); ("POUTOU", 111);
      ("DUPONT-AIGNAN", 341)]]]);
  N (Dpt "Isère",
    [Bv ("Meylan",
      [("ARTHAUD", 28); ("ROUSSEL", 169); ("MACRON", 4457); ("LASSALLE", 164);
      ("LE PEN", 1288); ("ZEMMOUR", 928); ("MÉLENCHON", 2198);
      ("HIDALGO", 251); ("JADOT", 906); ("PÉCRESSE", 763); ("POUTOU", 64);
      ("DUPONT-AIGNAN", 162)]]);
    Bv ("Echirolles",
      [("ARTHAUD", 104); ("ROUSSEL", 506); ("MACRON", 3276); ("LASSALLE", 259);
      ("LE PEN", 2737); ("ZEMMOUR", 779); ("MÉLENCHON", 5121);
      ("HIDALGO", 223); ("JADOT", 590); ("PÉCRESSE", 360); ("POUTOU", 92);
      ("DUPONT-AIGNAN", 202)]]);
    Bv ("Fontaine",
      [("ARTHAUD", 55); ("ROUSSEL", 363); ("MACRON", 2111); ("LASSALLE", 146);
      ("LE PEN", 1835); ("ZEMMOUR", 541); ("MÉLENCHON", 3113);
      ("HIDALGO", 185); ("JADOT", 493); ("PÉCRESSE", 212); ("POUTOU", 83);
      ("DUPONT-AIGNAN", 121)]]);
    Bv ("Saint-Martin-d'Hères",
      [("ARTHAUD", 58); ("ROUSSEL", 436); ("MACRON", 2769); ("LASSALLE", 207);
      ("LE PEN", 2289); ("ZEMMOUR", 661); ("MÉLENCHON", 4763);
      ("HIDALGO", 242); ("JADOT", 777); ("PÉCRESSE", 300); ("POUTOU", 119);
      ("DUPONT-AIGNAN", 161)]]);
    Bv ("Gières",
      [("ARTHAUD", 16); ("ROUSSEL", 66); ("MACRON", 1071); ("LASSALLE", 84);
      ("LE PEN", 641); ("ZEMMOUR", 205); ("MÉLENCHON", 844); ("HIDALGO", 96);
      ("JADOT", 301); ("PÉCRESSE", 155); ("POUTOU", 30);
      ("DUPONT-AIGNAN", 61)]]);
    Bv ("Grenoble",
      [("ARTHAUD", 256); ("ROUSSEL", 1300); ("MACRON", 15968);
      ("LASSALLE", 845); ("LE PEN", 6444); ("ZEMMOUR", 3389);
      ("MÉLENCHON", 24568); ("HIDALGO", 1488); ("JADOT", 5644);
      ("PÉCRESSE", 2019); ("POUTOU", 508); ("DUPONT-AIGNAN", 661)]]]]])

```

Question 22 En analysant soigneusement l'exemple précédent, donner les définitions des types `ville`, `zone` et `arbre`.

Question 23 Écrire une fonction `trouve_bv` qui extrait le résultat d'un bureau de vote à partir d'un arbre et du nom du bureau de vote.

```

# trouve_bv ara "Gières";;
- : depouillement =
[("ARTHAUD", 16); ("ROUSSEL", 66); ("MACRON", 1071); ("LASSALLE", 84);
("LE PEN", 641); ("ZEMMOUR", 205); ("MÉLENCHON", 844); ("HIDALGO", 96);
("JADOT", 301); ("PÉCRESSE", 155); ("POUTOU", 30); ("DUPONT-AIGNAN", 61)]

```

Question 24 À l'aide de la fonction `union` et de la fonction `trouve_bv` donner le résultat de la présidentielle 2022 si seulement les villes de Grenoble, Fontaine et Valence étaient prises en compte.

5 Conclusion

Conclure sur les défauts de la méthode du scrutin uninominal et ceux du jugement majoritaire.

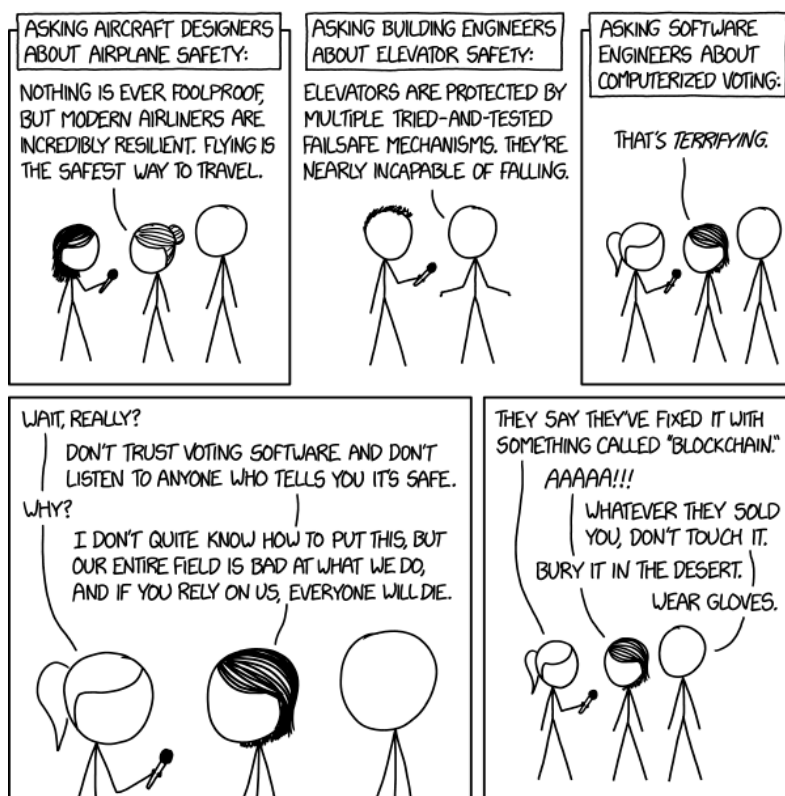


FIGURE 4 –

Références

- Majority Judgment, Measuring, Ranking and Electing, *Michel Balinski and Rida Laraki*, 2010
- Sur la forme des élections, *Nicolas De Condorcet*