Algorithmes pour flux de données

Claire Mathieu

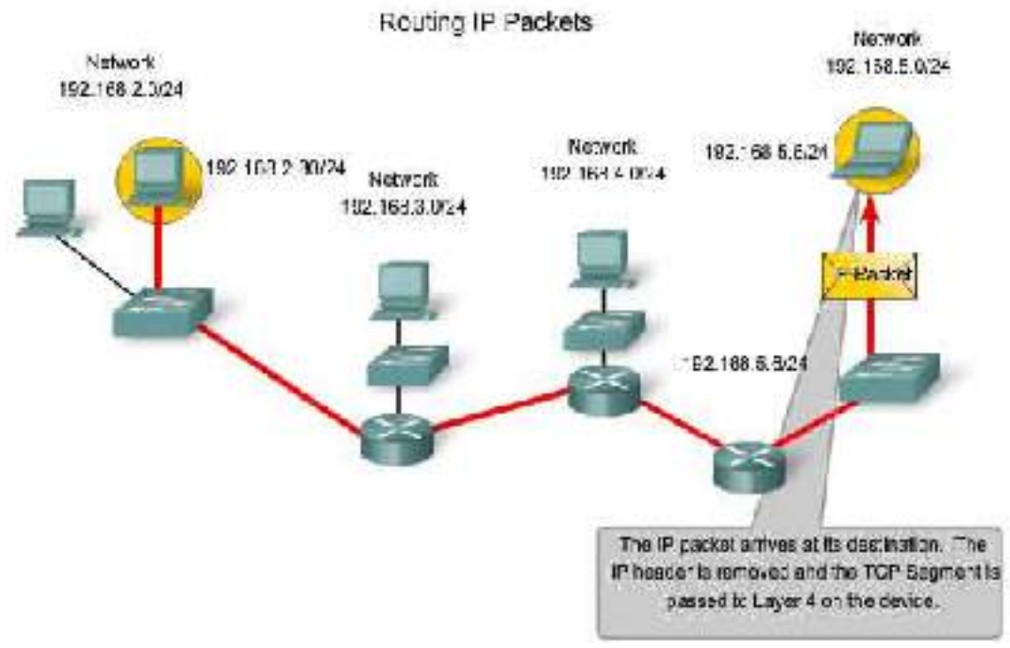
Compter les données distinctes
Vérifier le format d’un fichier xml
Faire un recrutement immédiat

Compter les données distinctes

Étude du trafic sur un réseau

Des personnes consultent un site web.
Combien y a-t-il de “visiteurs uniques” ?

Des paquets passent par un noeud.
On observe leurs adresses de destination.
Combien d’adresses distinctes ?



Attaques de type “déni de service”

Établissement d’une connexion téléphonique

Allô ?…

…Allô !

Centre d’appels téléphonique

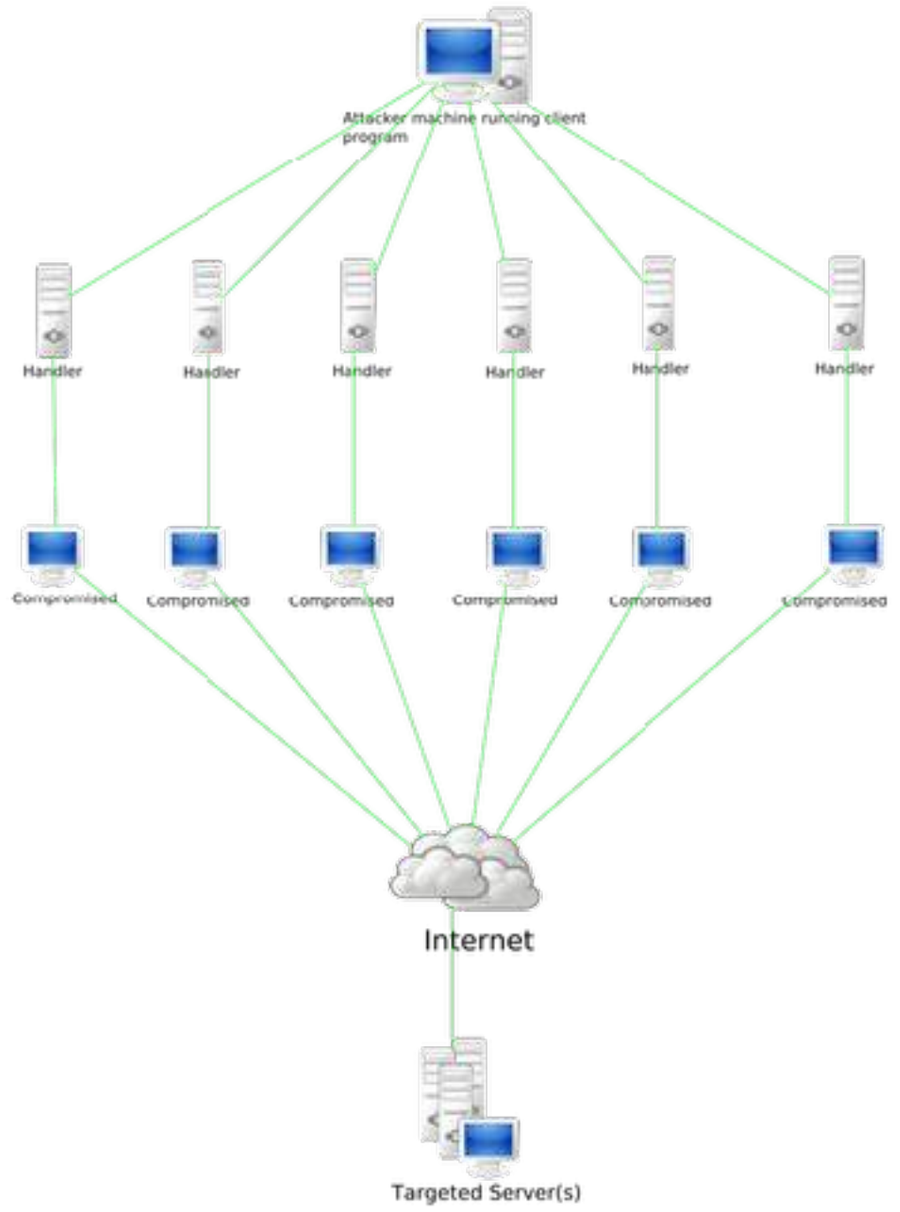
S’il y a trop d’appels
le centre d’appels est saturé
“Veuillez rappeler ultérieurement”

Comment saturer un centre d'appels

- Appeller le numéro
- Un opérateur décroche et dit : “Allô ?”
- Pendant qu’il attend la réponse, rappeler le même numéro
- Un autre opérateur décroche et dit : “Allô ?”
- Pendant qu’il attend, faire un troisième appel

Pendant que tous les opérateurs sont occupés à attendre une réponse à leur “ Allô ?” , ils ne peuvent pas prendre les appels légitimes.

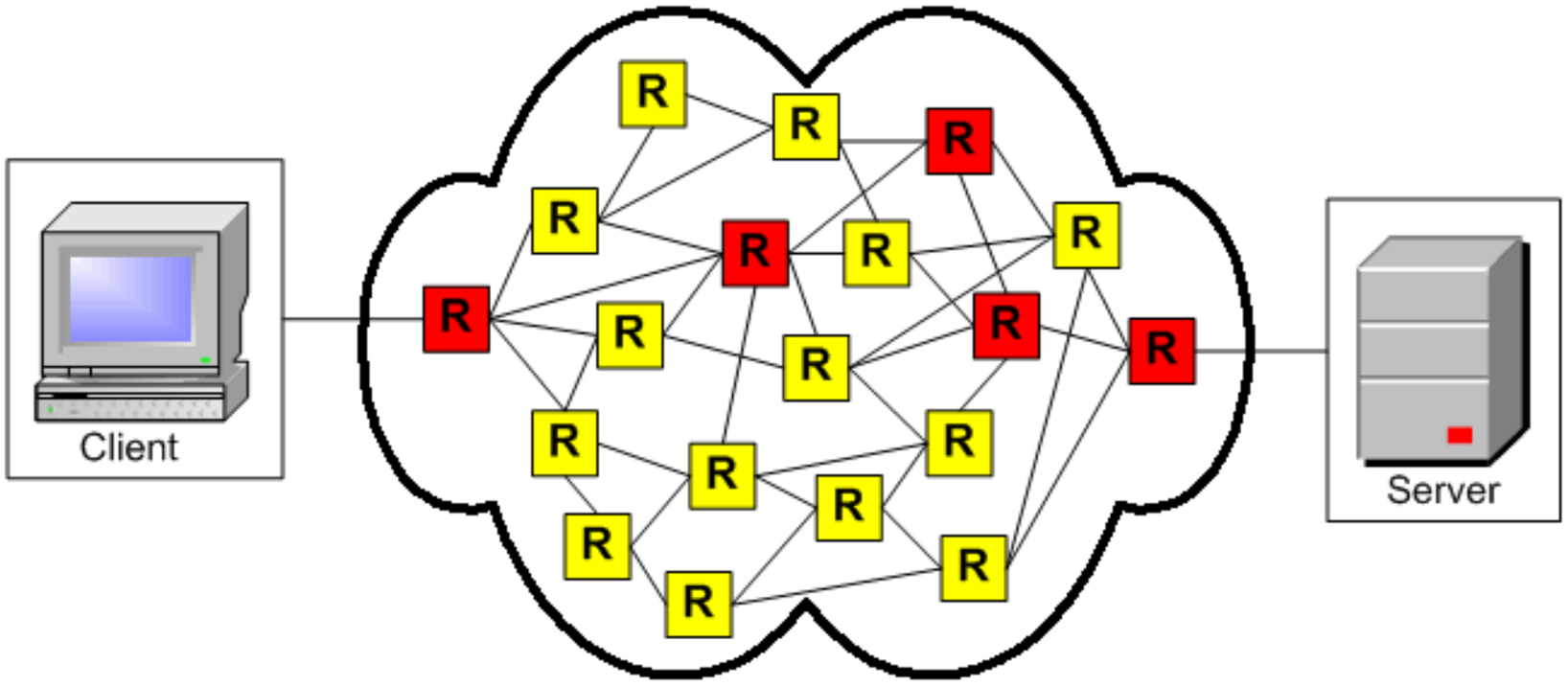
Une attaque de type “déni de service”



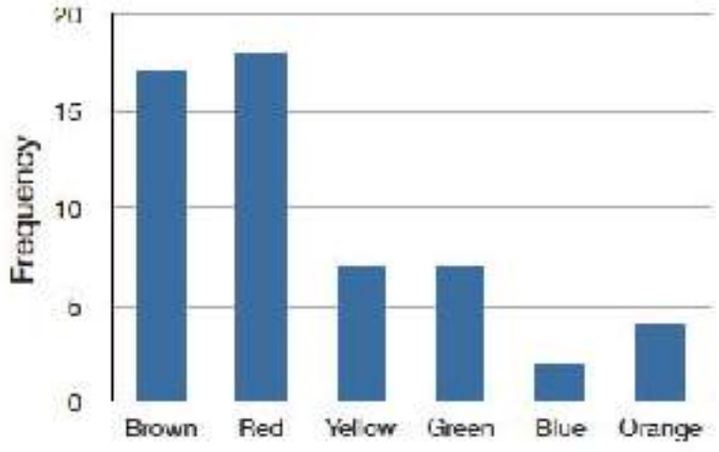
Serveur cible de l’attaque

L’attaquant crée artificiellement une congestion

Comment détecter ce type d’attaques ?



**Au lieu de simplement transmettre les paquets,
surveiller les phénomènes anormaux :
distribution des adresses de destination**



Gros volume
Grande rapidité
Petite mémoire

Flux de données

Suite d’objets a,b,c,a,d,e,f,a,…

Un problème : estimer combien il y a d’objets distincts

a,b,c,a,d,e,f,a,c : 6 objets distincts

Flux de données

Suite d'objets a,b,c,a,d,e,f,a,…

Problème :
combien d'objets distincts

a,b,c,a,d,e,f,a,c : 6 objets distincts

Stocker et compter : pas assez de place

But : estimation approximative mais rapide et prenant peu de place

Algorithme de Flajolet et Martin

Associer à chaque visiteur un entier
Compter le nombre de zéros terminant cet entier

Alice Dupont	**0111001001**	**0**
Barbara Durand	**1011011100**	**2**
Charlotte Martin	**1101110000**	**4**
Delphine Petit	**1101011110**	**1**
Ève Dubois	**1011011100**	**2**

Le premier nombre manquant est 3
2^3=8

Estimation : 8/c visiteurs uniques
(c=.77)

Associer à chaque visiteur un entier
Compter le nombre de zéros terminant cet entier

Alice Dupont	0111001001	0
Barbara Durand	1011011100	2
Charlotte Martin	1101100000	4
Delphine Petit	1101011110	1
Ève Dubois	1011011100	2

Le premier “trou” vaut 3
2^3=8
Estimation : 8/c visiteurs uniques

10111001
11011011
00101011
01110101
00110101
11110111
11111111

10101010
01110010
11001110
00101110

01110100
11010100

01101000

10110000

Analyse

Observation

Si Barbara Durand fait plusieurs visites :
ça ne change rien

Observation

S’il y a 16 visiteurs uniques:
Environ 8 entiers finissent par 1
et environ 8 finissent par 0.
Parmi ceux là, environ 4 finissent par 10
et environ 4 finissent par 00.
Parmi ceux là, environ 2 finissent par 100
et environ 2 finissent par 000
Parmi ceux là, environ 1 finit par 1000
et environ 1 finit par 0000
Celui là a une chance sur deux de finir par 00000
Donc le trou vaut à peu près 4
2^4=16 : gagné

Hypothèse : Les entiers sont aléatoires

Associer à chaque visiteur un entier
Compter le nombre de zéros terminant cet entier

Alice Dupont	0111001001	0
Barbara Durand	1011011100	2
Charlotte Martin	1101110000	4
Delphine Petit	1101011110	1
Ève Dubois	1011011100	2

Analyse

- Rapide
- Presque pas de place mémoire
- Pour que ce soit plus précis : répéter et prendre une moyenne de médianes

Le trou vaut 3, et 2^3=8
Estimation : 8/c visiteurs uniques

NB : La fonction : “ Alice Dupont” ⟶ 0111001001
doit se comporter comme une fonction aléatoire (hachage).

Estimation
du nombre de “visiteurs uniques”
avec précision de 5%
et presque pas d’espace mémoire utilisé

✓

Vérifier le format d’un fichier xml

Flux de données comme modèle de calcul

Suite d’objets a,b,c,a,d,e,f,a,…

Faire un calcul avec mémoire insuffisante pour stocker les données

Lire un fichier xml et vérifier qu'il est bien écrit

```
<?xml version="1.0" encoding="UTF-8" ?>
- <humain>
  - <corps>
      <tete>petite tete</tete>
    - <bras>
        <bras_1>bras_1</bras_1>
        <bras_2>bras_2</bras_2>
      </bras>
    - <jambes>
        <jambe_1>jambe_1</jambe_1>
        <jambe_2>jambe_2</jambe_2>
      </jambes>
    </corps>
  - <esprit>
    - <lob_droit>
        <equilibre>l'equilibre</equilibre>
      </lob_droit>
    - <lob_gauche>
        <la_parole>la parole</la_parole>
        <les_sentiments>les_sentiments</les_sentiments>
      </lob_gauche>
      <pensee>la pensee</pensee>
    </esprit>
</humain>
```

Abstraction du problème

Suite de parenthèses et de crochets.

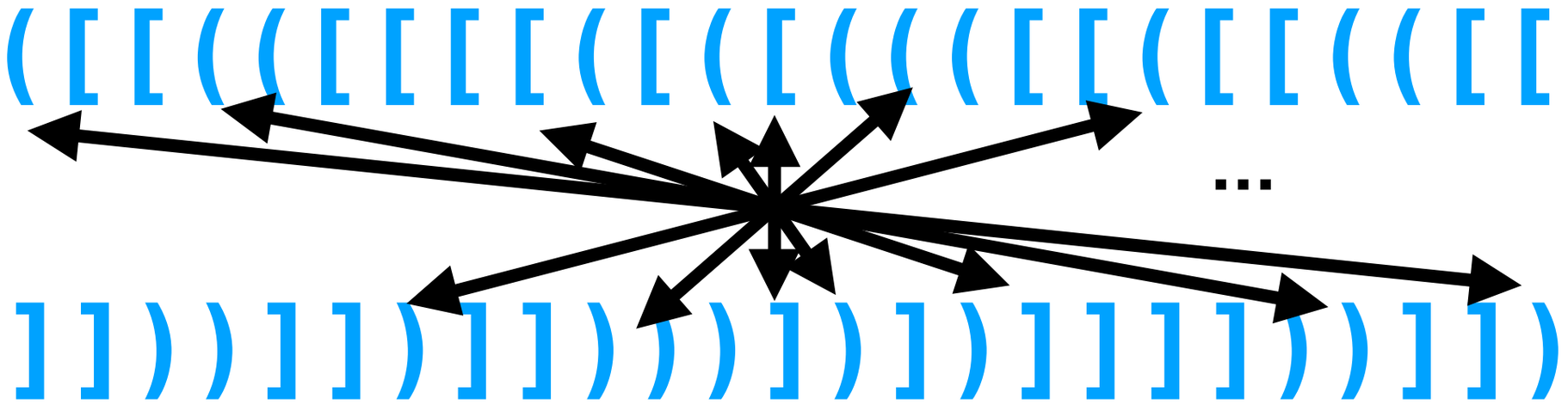
([[(([ ]))]])

Est-elle bien formée ?

([[(([ ]))]])

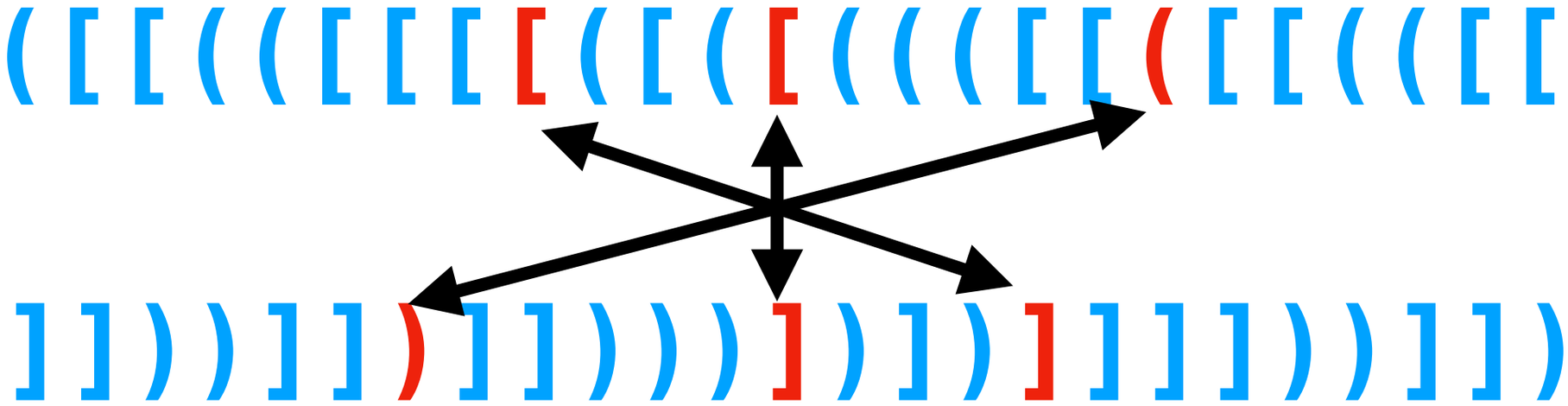
([(([ ]))]])

Comment tester cela ?



Empiler toutes les parenthèses et crochets ouvrants
Vérifier que chaque parenthèse ou crochet fermant correspond

Trop gourmand en mémoire



Choisir quelques positions : 9, 13, 19
Empiler les parenthèses et crochets ouvrants dans ces positions
Vérifier que chaque parenthèse ou crochet fermant dans les positions associées est du type correspondant

Trop imprécis
Ne détectera pas une erreur

Une solution économe et fiable

À chaque parenthèse on associe une “hauteur”

- **une parenthèse ouvrante augmente la hauteur de 1**
- **une parenthèse fermante diminue la hauteur de 1**

([[(([[[[([([((([[([[(([[

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

]]))]])]])))])])]]]]))]])

25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01

On choisit un nombre premier p valant environ 10n.
À chaque parenthèse à hauteur h on associe un nombre :

$$
\begin{aligned}
\textbf{(} &\mapsto 3^h \pmod p \\
\textbf{)} &\mapsto -3^h \pmod p \\
\textbf{[} &\mapsto 0 \pmod p \\
\textbf{]} &\mapsto 0 \pmod p
\end{aligned}
$$

On fait la somme :

$$3^1 + 3^4 + 3^5 + 4^{10} + 3^{12} + 3^{14} + 3^{15} + 3^{16} + 3^{19} + 3^{22} + 3^{23}$$
$$-3^{23} - 3^{22} - 3^{19} - 3^{16} - 3^{15} - 3^{14} - 3^{12} - 3^{10} - 3^5 - 3^4 - 3^1 = 0$$

Si la suite de parenthèses est bien formée, la somme vaut 0.

Si la suite de parenthèses est mal formée, la somme peut valoir 0 par accident :

([[([[[[[([((((([[([[(([[

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

]]))]])]])))])])]]]]))]])

25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01

Il peut se trouver, par accident, que $3^{13} - 3^5 = 0 \pmod{p}$

Idée : au lieu de 3, prendre une variable x entre 0 et p-1
La suite de parenthèses donne une somme qui est un polynôme en x, Q(x), de degré au plus n

Un polynôme Q(x) non nul et de degré au plus n a au plus n racines Q(x)=0
p=10n
Si x est choisi au hasard entre 0 et p-1 alors Pr(Q(x)=0) < 1/10

Algorithme

On choisit un nombre premier p valant environ 10n.
On choisit un nombre x au hasard entre 0 et p-1.
À chaque parenthèse on associe une “hauteur”
- **une parenthèse ouvrante augmente la hauteur de 1**
- **une parenthèse fermante diminue la hauteur de 1**
À chaque parenthèse à hauteur h on associe un nombre :

($\mapsto x^h \pmod p$

) $\mapsto -x^h \pmod p$

[$\mapsto 0 \pmod p$

] $\mapsto 0 \pmod p$

On fait la somme.
Si elle ne vaut pas zéro, on dit : “la suite n’est pas bien formée”
Si elle vaut zéro, on dit : “la suite est probablement bien formée”

- **Possibilité de faire des calculs sur des données même sans place pour les stocker**
- **Puissance des techniques probabilistes**
- **Utilité de traduction dans le monde algébrique**

Faire un recrutement immédiat

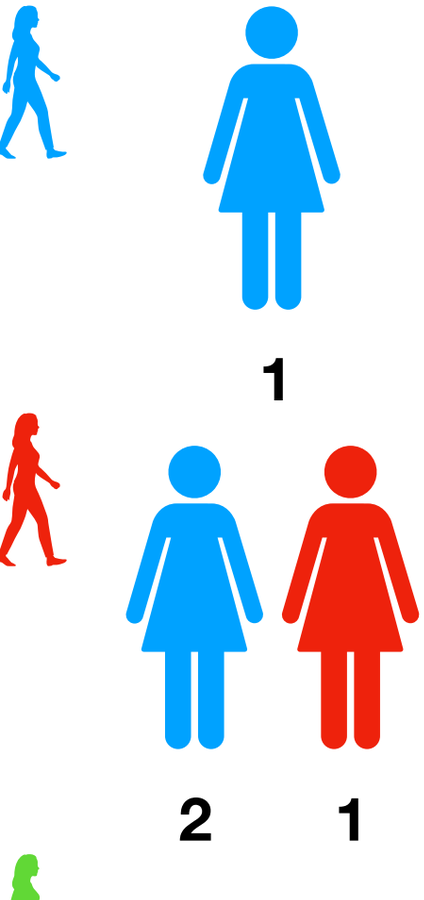
Candidats à un poste

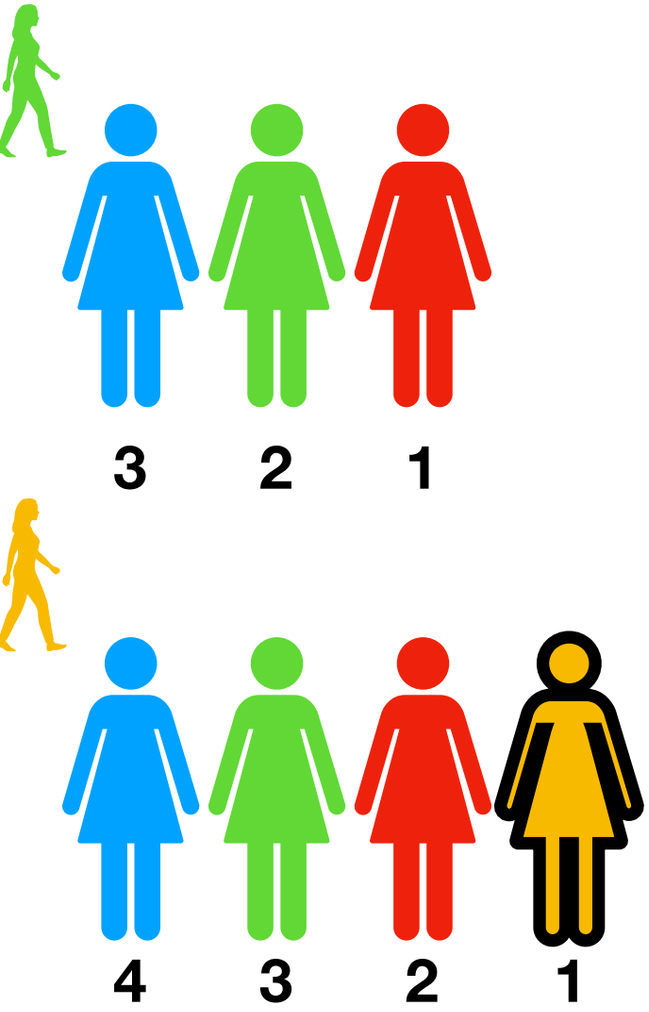
Recrutement “en ligne” (immédiat)

100 candidats

Entrevues avec les candidats : un par un
Après avoir vu un candidat :
on sait si c’est le meilleur
de tous ceux qu’on a vus jusqu’à présent

On peut lui offrir le poste
ou lui dire : “désolé, votre candidature ne nous intéresse pas”
et interviewer le candidat suivant





Quel est le but ?

Idéalement : recruter le meilleur candidat

Mais… impossible d'attendre — on doit décider si on fait une offre à un candidat au moment où il est là

Quelles informations nous donne un entretien ?

Le classement du candidat relativement à ceux qu'on a déjà vus

NB -Le premier entretien ne donne aucune information

Si l'algorithme recrute le premier candidat…
Ce pourrait être le pire de tous !
Si l'algorithme ne recrute pas le premier candidat…
Ce pourrait être le meilleur de tous !
Et alors,
Si l'algorithme recrute le deuxième candidat…
Ce pourrait être le pire de tous !
Et sinon…
Ce pourrait être le deuxième meilleur !

Théorème

Quelle que soit la stratégie
On a peu de chances de se retrouver avec le meilleur candidat

Hypothèse supplémentaire :
les candidats arrivent dans un ordre aléatoire

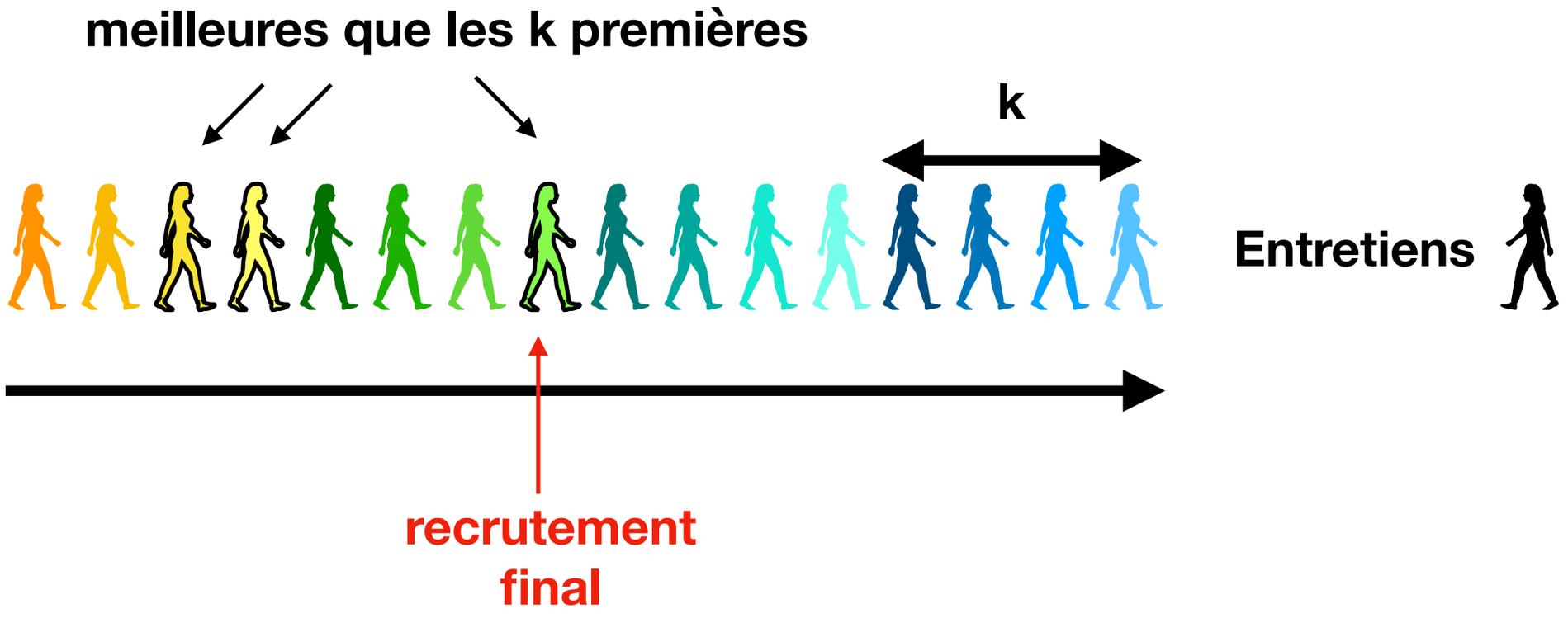
Algorithme pour N candidats : Réflexion, Action

Algorithme Réflexion-Action

k=N/e

Interviewer les k premiers candidats sans leur faire d'offre

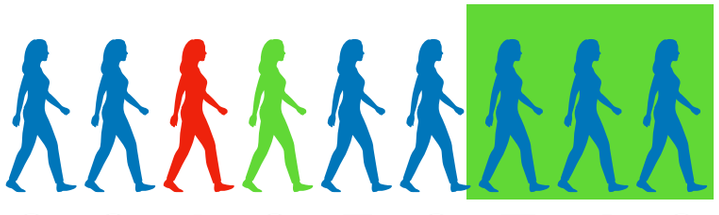
Faire une offre au premier candidat meilleur que les k premiers candidats



Analyse : structure

La meilleure candidate sera-t-elle recrutée ?

- **si elle se trouve parmi les k premiers interviewés, c'est perdu**
- **sinon : elle est en position j, et alors :**
- **si la meilleure de celles qui sont en position 1,2,…,j-1 se trouve parmi les k premiers interviewés, c'est gagné, sinon, c'est perdu**



8, 2, 1, 3, 5, 9, 7, 4, 6

Celle qu'on aurait voulue **Celle qu'on recrute**

Analyse : calcul

Probabilité que la meilleure soit en position j : 1/N
Probabilité que la meilleure parmi celles en positions 1,2,…,j-1 soit parmi les k premiers : k/(j-1)
Probabilité que la meilleure soit recrutée :

$$\sum_{k+1}^{n} \frac{1}{N} \frac{k}{j-1} \sim \frac{1}{e} \sim 37\%$$

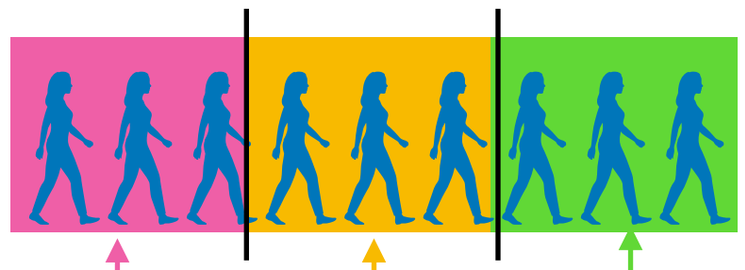
C'est le mieux qu'on puisse faire

Généralisation pour recruter une équipe de 3 personnes

Hypothèse : sous-modularité =
Plus on a déjà recruté de monde,
moins le recrutement d'Alice apporte de valeur ajoutée.

Valeur(Alice)
> Valeur(Alice,Barbara)-Valeur(Barbara)
> Valeur(Alice, Barbara, Charlotte)-Valeur(Barbara, Charlotte)

Faire 3 groupes de N/3 candidats
Dans chaque groupe
  Interviewer les k=(N/3)/e premiers candidats sans leur faire d'offre
  Faire une offre au premier candidat du groupe dont la valeur marginale
    est meilleure que celle des k premiers candidats



On choisit le premier membre de l'équipe

On choisit le deuxième membre de l'équipe

On choisit le troisième membre de l'équipe

Différences entre les modèles

Flux de données

Modèle en-ligne

Les données arrivent une par une au fil du temps

mémoire limitée **temps de calcul limité**	**mémoire illimitée** **temps de calcul illimité**
Résultat donné à la fin, **étapes intermédiaires** **arbitraires**	**Décision prise à chaque instant** **irrévocable** **Résultat : accumulation des décisions**
Surveillance : se passe-t-il **quelque chose d'anormal ?** **Maintenir une liste** **de bons itinéraires** **en fonction des infos trafic**	**Choisir quelle publicité afficher** **à côté d'un résultat de recherche** **Répondre à une offre, achat/vente** **Choisir un itinéraire**

On ne connaît pas l'avenir

Conclusion

Modèles d’accès aux données

- un monde dynamique et éphémère
- estimations approchées

Omniprésence des probabilités en algorithmique

- Techniques de conception d’algorithme
- Hypothèses stochastiques, techniques d'analyse