Master 2 Informatique Fondamentale et Appliquée Année 2022-2023

Amélie Gheerbrant, Arnaud Sangnier, Ralf Treinen

02/12/2022

Calendrier 22/23

- Deuxième période : du 02/01 au 24/03 (cours/TD/TP)
  Examens : 27/03 au 31/03
- Troisième période : du 01/04 au 30/09 (stage)
  Soutenance de stage, et jury du M2 : mi-septembre

Rappel : Vous devrez valider 60 crédits

- 11 modules d'informatique (chacun à 3 crédits), modules à choisir selon les règles spécifiques de votre parcours.

Rappel : Vous devrez valider 60 crédits

- 11 modules d'informatique (chacun à 3 crédits), modules à choisir selon les règles spécifiques de votre parcours.
- le module d'Anglais, première période (3 crédits)

Rappel : Vous devrez valider 60 crédits

- 11 modules d'informatique (chacun à 3 crédits), modules à choisir selon les règles spécifiques de votre parcours.
- le module d'Anglais, première période (3 crédits)
- le stage en entreprise pendant la troisième période (24 crédits)

Rappel : Vous devrez valider 60 crédits

- 11 modules d'informatique (chacun à 3 crédits), modules à choisir selon les règles spécifiques de votre parcours.
- le module d'Anglais, première période (3 crédits)
- le stage en entreprise pendant la troisième période (24 crédits)
- Dans les trois parcours on peut aussi choisir un cours "externe" (par ex au MPRI, LMFI, une autre université) *avec accord du responsable de parcours*.

Les règles du parcours DATA

Anglais + 11 cours d'Info, dont au moins 7 *cours fondamentaux* :

1ère période	2ème période
Architecture des Systèmes de Bases de Données	Architecture des Systèmes d'Information
Base de données spécialisées	Algorithmique Répartie
Fouille de Données et Aide à la Décision	Méthodes Algorithmiques pour l'Accès à l'Information Numérique
Programmation Objets : Concepts Avancées	Programmation Logique et Par Contraintes Avancée
	Grands Réseaux d'Interaction
	Optimisation
	Programmation Répartie
	Introduction Apprentissage Profond

Les règles du parcours IMPAIRS

Anglais + 11 cours d'Info, dont au moins 8 *cours suggérés* :

1ère période	2ème période
Architecture des Systèmes de Bases de Données	Architecture des Systèmes d'Information
Fouille de données et aide à la décision	Méthodes algorithmiques pour l'accès à l'information numérique
Informatique embarquée	Administration système et réseaux
Ingénierie des protocoles	Algorithmique répartie
Modélisation et Spécification	Programmation répartie
Programmation Synchrone	Grands réseaux d'interaction
Protocoles des services Internet	Mobilité

Les règles du parcours LP

Anglais + 11 cours d'Info, dont 3 *cours obligatoires* :

1ère période	2ème période
Programmation Objets : Concepts Avancées	Programmation Comparée
	Transformation de Programmes

et au moins 5 parmi les *cours recommandés* :

1ère période	2ème période
Architecture des Systèmes BD	Architecture des SI
Méthodes Formelles de Vérification	Programmation Logique et Par Contraintes Avancée
Modélisation et Spécification	Programmation Répartie
Programmation Synchrone	Analyse Statique

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé
  - Interfaces et Outils de MacOS-X

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé
  - Interfaces et Outils de MacOS-X
- En contrepartie :

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé
  - Interfaces et Outils de MacOS-X
- En contrepartie :
  - *Grands Réseaux d'Interaction* renforcé par un deuxième groupe de TD

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé
  - Interfaces et Outils de MacOS-X
- En contrepartie :
  - *Grands Réseaux d'Interaction* renforcé par un deuxième groupe de TD
  - *Introduction à l'Apprentissage Profond* renforcé par un deuxième groupe de TD

Changements par rapport à la maquette/l'année dernière

- Deux cours n'ont pas lieu cette année :
  - Systèmes Avancé
  - Interfaces et Outils de MacOS-X
- En contrepartie :
  - *Grands Réseaux d'Interaction* renforcé par un deuxième groupe de TD
  - *Introduction à l'Apprentissage Profond* renforcé par un deuxième groupe de TD
  - Deuxième édition de *POCA*

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.
- Vous choisissez 6 cours d'informatique pour la deuxième période (4 pour les LP car il y a déjà 2 obligatoires).

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.
- Vous choisissez 6 cours d'informatique pour la deuxième période (4 pour les LP car il y a déjà 2 obligatoires).
- Si vous souhaitez un cours en plus : envoyer un mail à votre responsable de parcours et on va essayer de vous y inscrire (sans promesse).

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.
- Vous choisissez 6 cours d'informatique pour la deuxième période (4 pour les LP car il y a déjà 2 obligatoires).
- Si vous souhaitez un cours en plus : envoyer un mail à votre responsable de parcours et on va essayer de vous y inscrire (sans promesse).
- Il n'est pas permis de vous inscrire à des cours qui ont lieu au même moment.

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.
- Vous choisissez 6 cours d'informatique pour la deuxième période (4 pour les LP car il y a déjà 2 obligatoires).
- Si vous souhaitez un cours en plus : envoyer un mail à votre responsable de parcours et on va essayer de vous y inscrire (sans promesse).
- Il n'est pas permis de vous inscrire à des cours qui ont lieu au même moment.
- Pour la deuxième période : du 2/12 le soir au 9/12.

Inscriptions pédagogiques à des modules

- Vous devrez choisir sur silice les cours que vous voulez suivre.
- Vous choisissez 6 cours d'informatique pour la deuxième période (4 pour les LP car il y a déjà 2 obligatoires).
- Si vous souhaitez un cours en plus : envoyer un mail à votre responsable de parcours et on va essayer de vous y inscrire (sans promesse).
- Il n'est pas permis de vous inscrire à des cours qui ont lieu au même moment.
- Pour la deuxième période : du 2/12 le soir au 9/12.
- Les cours ont des capacités limitées. La priorité sera donnée aux étudiants pour qui le cours demandé est recommandé.

Le Stage

- Durée minimale de 4 mois.

Le Stage

- Durée minimale de 4 mois.
- Durée maximale de 6 mois (contrainte légale)

Le Stage

- Durée minimale de 4 mois.
- Durée maximale de 6 mois (contrainte légale)
- Entre le 3 avril et le 30 septembre.

Le Stage

- Durée minimale de 4 mois.
- Durée maximale de 6 mois (contrainte légale)
- Entre le 3 avril et le 30 septembre.
- Rapport de stage, soutenance (normalement publique) en septembre.

Le Stage

- Durée minimale de 4 mois.
- Durée maximale de 6 mois (contrainte légale)
- Entre le 3 avril et le 30 septembre.
- Rapport de stage, soutenance (normalement publique) en septembre.
- Stage en entreprise (aussi startup) ou Stage de recherche.

Le Stage

- Durée minimale de 4 mois.
- Durée maximale de 6 mois (contrainte légale)
- Entre le 3 avril et le 30 septembre.
- Rapport de stage, soutenance (normalement publique) en septembre.
- Stage en entreprise (aussi startup) ou Stage de recherche.
- Stage à l'étranger possible, mais procédure administrative particulière.

Trouver son Stage

- **Le sujet doit être approuvé par le responsable du parcours.**

Trouver son Stage

- **Le sujet doit être approuvé par le responsable du parcours.**
- Rédaction et signature de la convention : voir `http://www.informatique.univ-paris-diderot.fr/formations/masters/m2_stages_fin_etudes`

Trouver son Stage

- **Le sujet doit être approuvé par le responsable du parcours.**
- Rédaction et signature de la convention : voir `http://www.informatique.univ-paris-diderot.fr/formations/masters/m2_stages_fin_etudes`
- Permettez du temps pour la signature de la convention (en particulier pour des organismes sensibles à la sécurité).

Trouver son Stage

- **Le sujet doit être approuvé par le responsable du parcours.**
- Rédaction et signature de la convention : voir `http://www.informatique.univ-paris-diderot.fr/formations/masters/m2_stages_fin_etudes`
- Permettez du temps pour la signature de la convention (en particulier pour des organismes sensibles à la sécurité).
- Commencez tôt (pendant la première période) à vous en occuper.

Trouver son Stage

- **Le sujet doit être approuvé par le responsable du parcours.**
- Rédaction et signature de la convention : voir `http://www.informatique.univ-paris-diderot.fr/formations/masters/m2_stages_fin_etudes`
- Permettez du temps pour la signature de la convention (en particulier pour des organismes sensibles à la sécurité).
- Commencez tôt (pendant la première période) à vous en occuper.
- Nous transmettons des offres de stages sur la liste de diffusion des parcours concernés ; vérifiez votre adresse email renseignée sur silice.

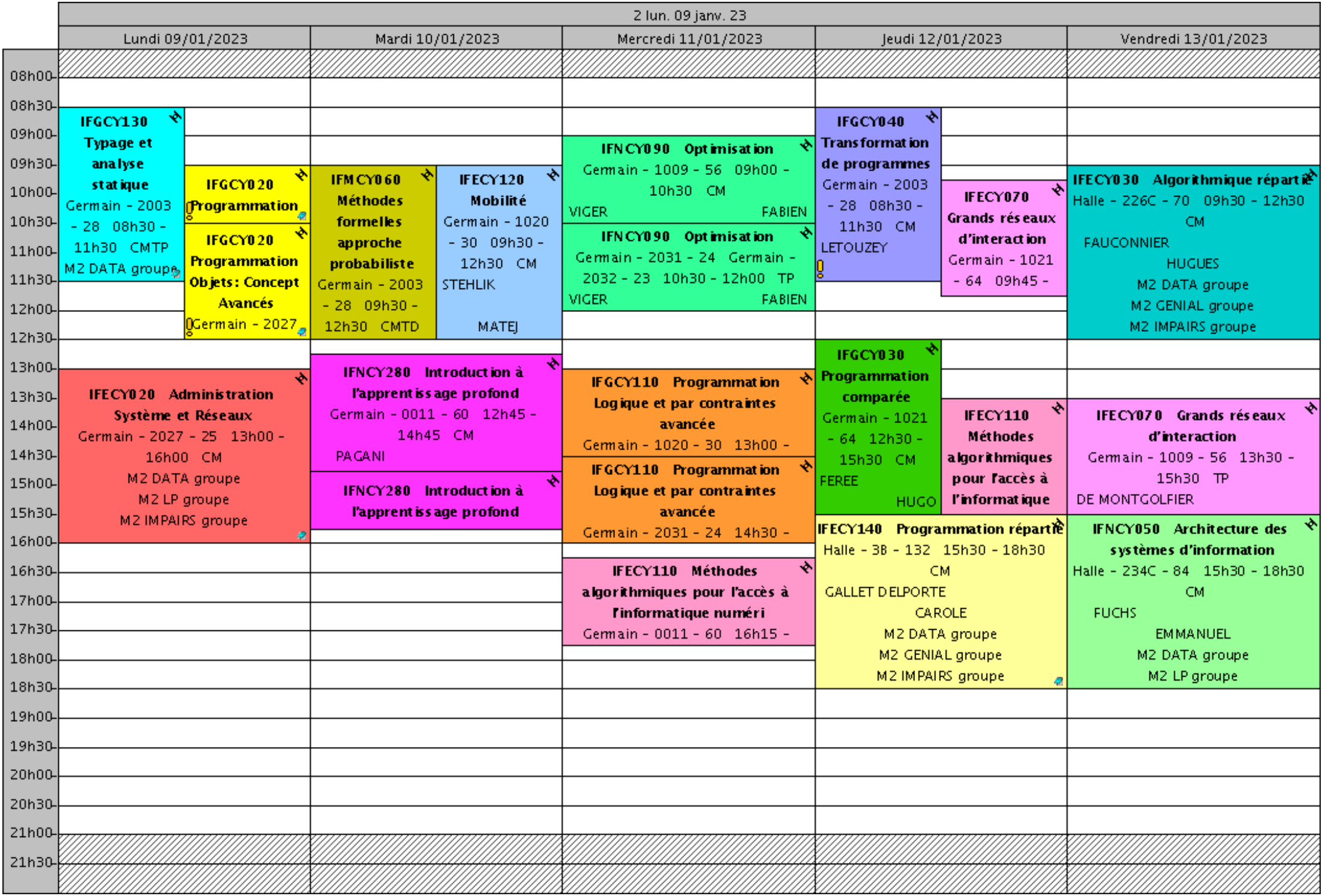
Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, bâtiment SG, `crochet@informatique.univ-paris-diderot.fr`
- Responsables de parcours : prendre rendez-vous par email
  - DATA : Amélie Gheerbrant `Amelie.Gheerbrant@irif.fr`
  - IMPAIRS : Arnaud Sangnier `Arnaud.Sangnier@irif.fr`
  - LP : Ralf Treinen `treinen@irif.fr`
- Ingénieur : Laurent Pietroni `pietroni@informatique.univ-paris-diderot.fr`

Emploi du temps

- Il est indicatif, l'edt définitif sera publié sur le wiki de l'UFR (catégorie "Plannings et Examens").

M2

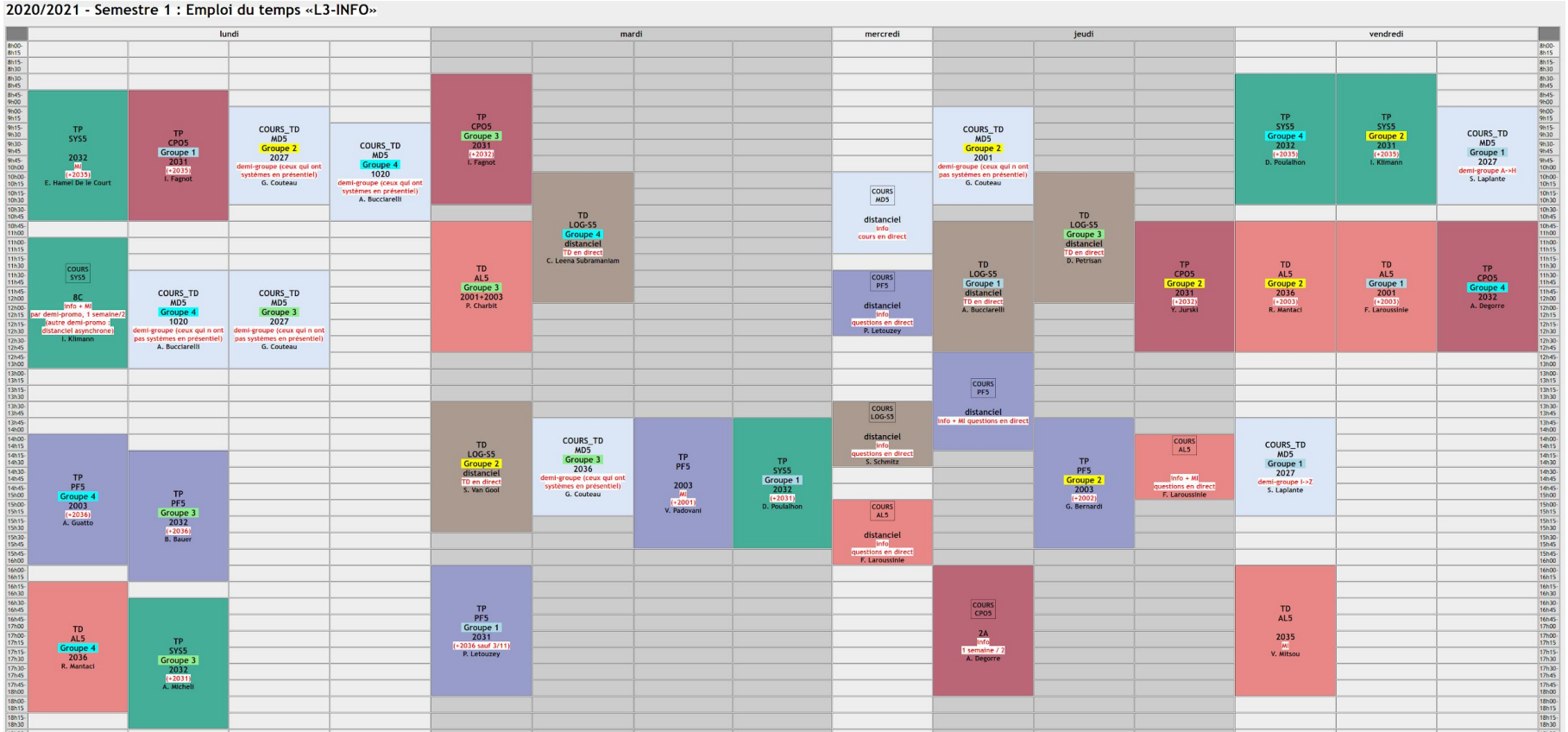

2 lun. 09 janv. 23
Lundi 09/01/2023
Mardi 10/01/2023
Mercredi 11/01/2023
Jeudi 12/01/2023
Vendredi 13/01/2023
08h00
08h30
09h00
09h30
10h00
10h30
11h00
11h30
12h00
12h30
13h00
13h30
14h00
14h30
15h00
15h30
16h00
16h30
17h00
17h30
18h00
18h30
19h00
19h30
20h00
20h30
21h00
21h30
IFGCY130 Typage et analyse statique
Germain - 2003 - 28 08h30 - 11h30 CMTP
M2 DATA groupe
IFGCY020 Programmation
IFGCY020 Programmation Objets: Concept Avancés
Germain - 2027
IFMCY060 Méthodes formelles approche probabiliste
Germain - 2003 - 28 09h30 - 12h30 CMTD
IFECY120 Mobilité
Germain - 1020 - 30 09h30 - 12h30 CM
STEHLIK
MATEJ
IFNCY090 Optimisation
Germain - 1009 - 56 09h00 - 10h30 CM
VIGER FABIEN
IFNCY090 Optimisation
Germain - 2031 - 24 Germain - 2032 - 23 10h30 - 12h00 TP
VIGER FABIEN
IFGCY040 Transformation de programmes
Germain - 2003 - 28 08h30 - 11h30 CM
LETOUZEY
IFECY070 Grands réseaux d'interaction
Germain - 1021 - 64 09h45 -
IFECY030 Algorithmique répartie
Halle - 226C - 70 09h30 - 12h30 CM
FAUCONNIER
HUGUES
M2 DATA groupe
M2 GENIAL groupe
M2 IMPAIRS groupe
IFECY020 Administration Système et Réseaux
Germain - 2027 - 25 13h00 - 16h00 CM
M2 DATA groupe
M2 LP groupe
M2 IMPAIRS groupe
IFNCY280 Introduction à l'apprentissage profond
Germain - 0011 - 60 12h45 - 14h45 CM
PAGANI
IFNCY280 Introduction à l'apprentissage profond
IFGCY110 Programmation Logique et par contraintes avancée
Germain - 1020 - 30 13h00 -
IFGCY110 Programmation Logique et par contraintes avancée
Germain - 2031 - 24 14h30 -
IFECY110 Méthodes algorithmiques pour l'accès à l'informatique numéri
Germain - 0011 - 60 16h15 -
IFGCY030 Programmation comparée
Germain - 1021 - 64 12h30 - 15h30 CM
FEREE
HUGO
IFECY110 Méthodes algorithmiques pour l'accès à l'informatique
IFECY070 Grands réseaux d'interaction
Germain - 1009 - 56 13h30 - 15h30 TP
DE MONTGOLFIER
IFECY140 Programmation répartie
Halle - 38 - 132 15h30 - 18h30 CM
GALLET DELPORTE
CAROLE
M2 DATA groupe
M2 GENIAL groupe
M2 IMPAIRS groupe
IFNCY050 Architecture des systèmes d'information
Halle - 234C - 84 15h30 - 18h30 CM
FUCHS
EMMANUEL
M2 DATA groupe
M2 LP groupe

Optimisation Combinatoire

http://fabien.viger.free.fr/oc

Qu’est-ce que l’Optimisation Combinatoire ?

Exemple: Création des emploi du temps



Qu’est-ce que l’Optimisation Combinatoire ?

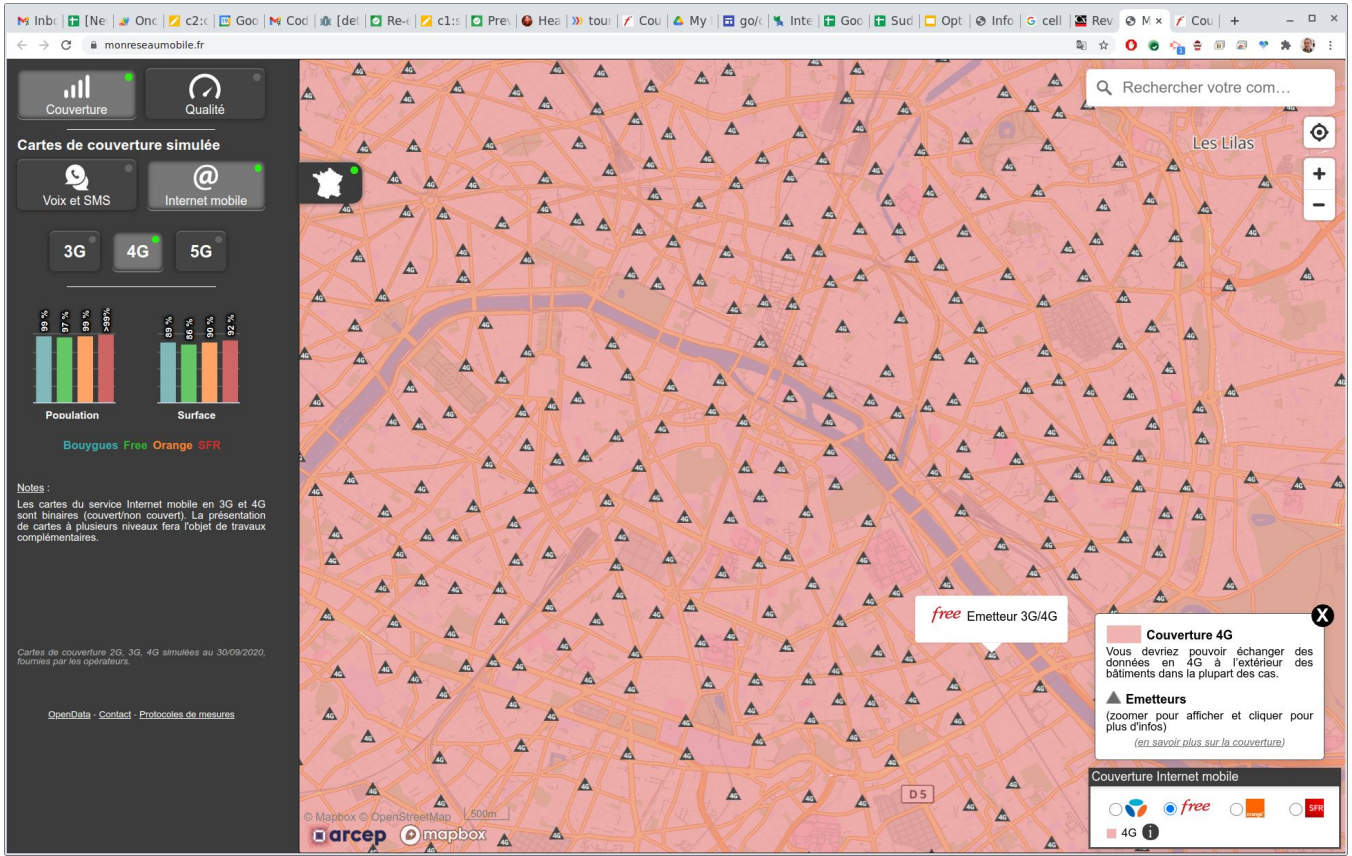
Autre exemple: Sudoku (cliquer sur l’image)

Input Grid

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

Qu’est-ce que l’Optimisation Combinatoire ?

Autre exemple: où placer les émetteurs 4G ?



Organisation du Cours

Mercredi 8h45-12h, Sophie Germain: 1h30 cours + 1h45 TD + commence la 2ème semaine (le 10 janvier).

Note = **50% Examen Final** (sur papier) et **50% TD**

Cours au tableau : pas de support de cours, mais vidéos dispo après chaque cours pour réviser.

TD en C++, avec **Auto-Tests** et corrigés dispos après le TD.

- Adaptés aux débutants en C++ (mais aux experts aussi)
- TD = application *directe* du cours.

Plan indicatif du Cours

1. **Graphes**. Implémentations (C++). Composantes [fortement] connexes. Tables de hachage. Parcours (BFS, DFS). Dijkstra.
2. **Arbres** et leurs algorithmes. Autres graphes particuliers (DAG, **Tri topologique**). Programmation dynamique: 1er exemple.
3. **Programmation dynamique**: autres exemples.
4. **Algorithmes Gloutons:** Arbre couvrant minimum et autres.
5. **Heuristiques, Monte-Carlo** (exemples: diamètre d'un graphe. Miller-Rabin).
6. **Programmation linéaire** (*aka* LP)
7. Programmation linéaire **entière** (*aka* **MIP**)
8. MIP: Modélisation avancée.
9. **Examen blanc** commenté

Extensions possible du Cours

- La revolution **SAT** (CDCL)
- **Max-Flow**, Min-cost flow
- Revisions d'algorithmique: Hash tables. Tas. Complexité.

Tout est dispo sur :

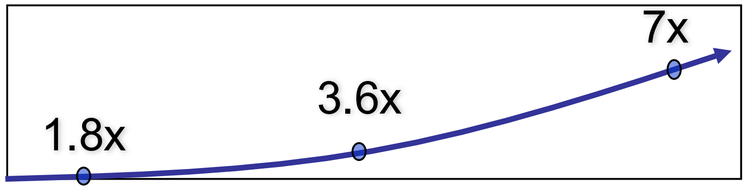
http://fabien.viger.free.fr/oc

Programmation répartie

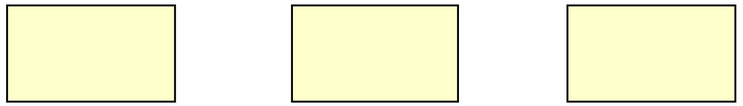
Jeudi 15h30-18h30

Carole Delporte cd@irif.fr

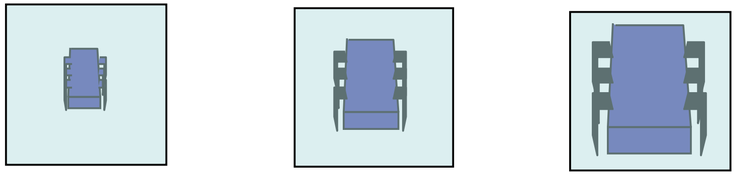
Accélaration



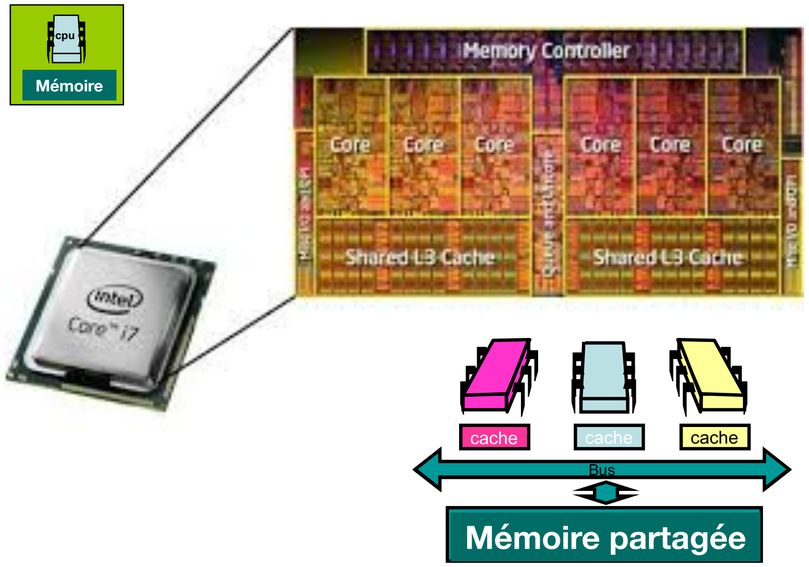
Code utilisateur

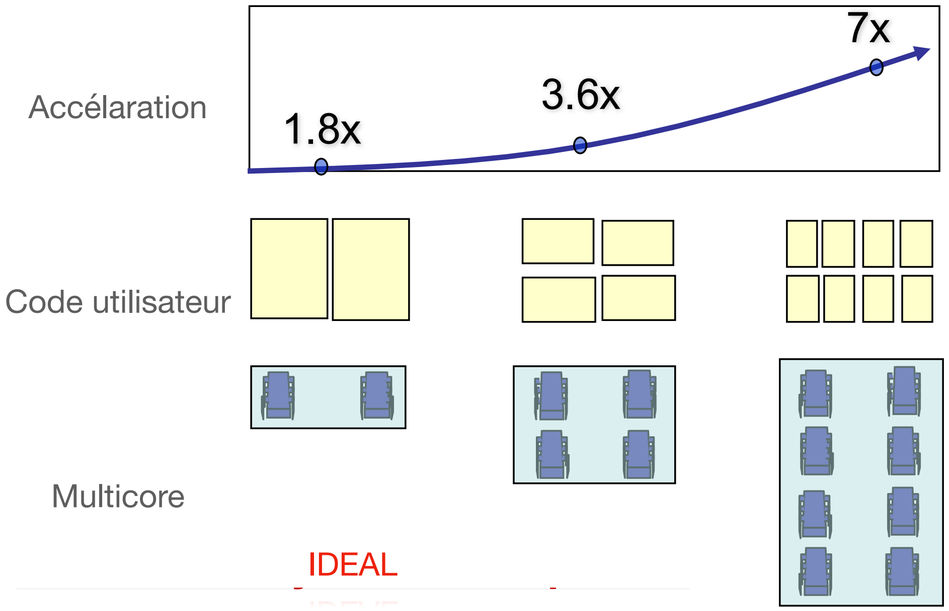


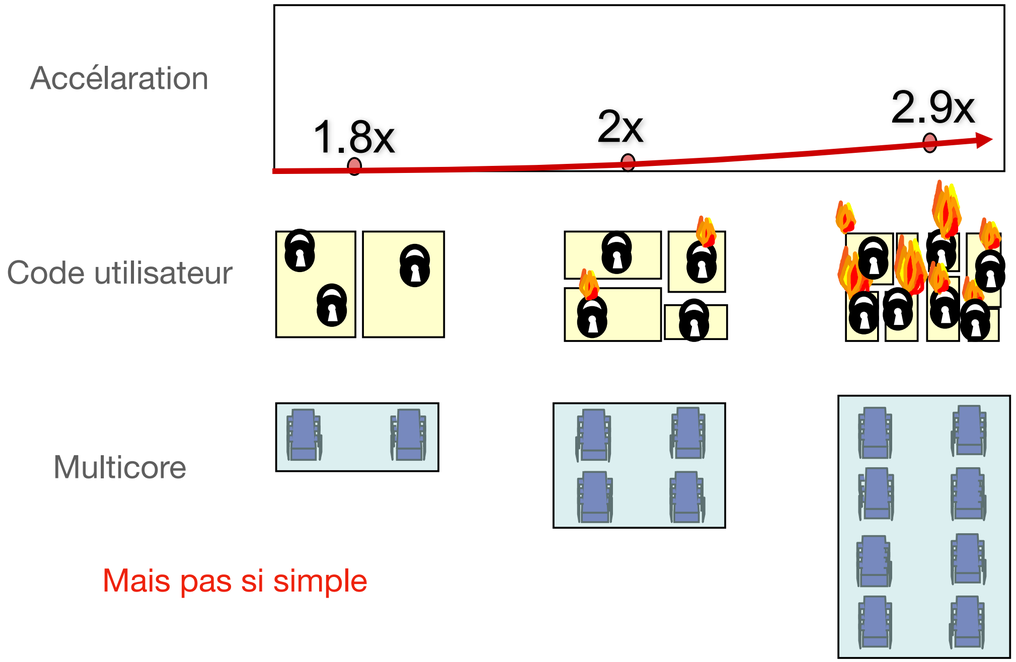
Multicore



Temps : Loi de Moore







Apprenez les principes fondamentaux de la programmation de plusieurs threads accédant à la mémoire partagée des simples verrous aux primitives plus évoluées

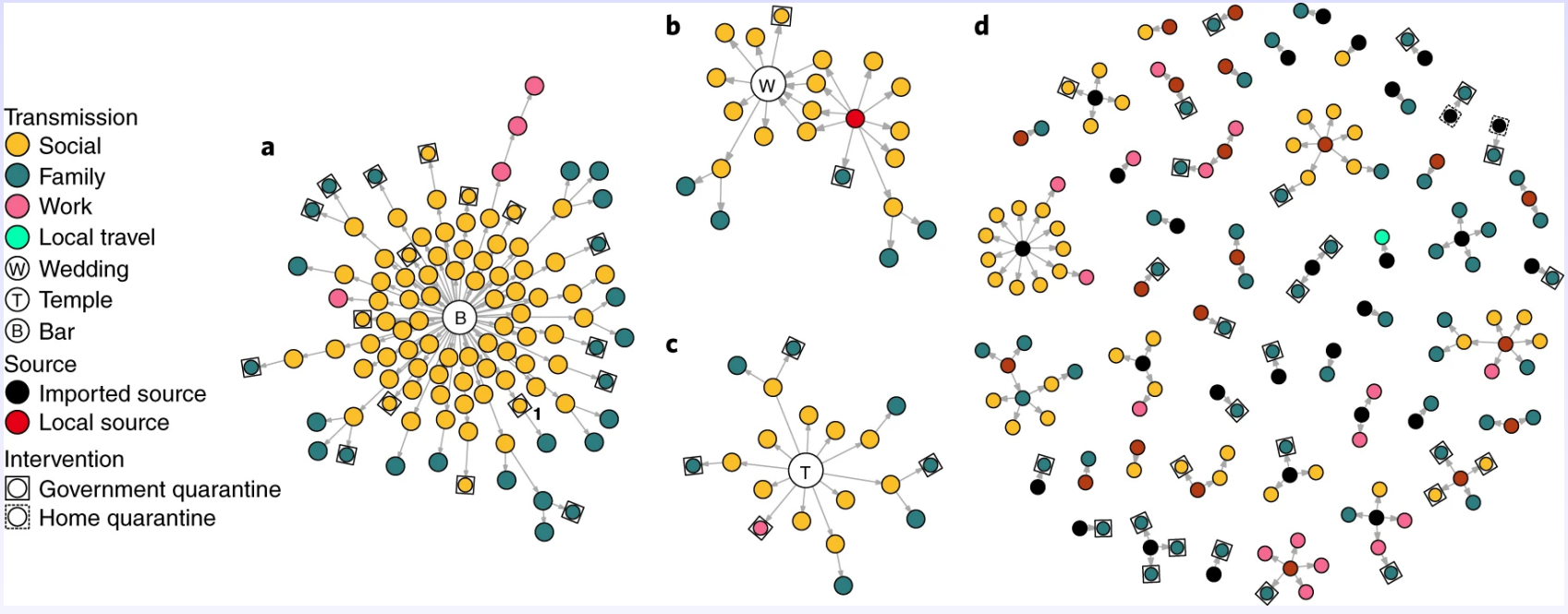
Programmez (en Java) des structures de données concurrentes

Grands Réseaux d'Interaction

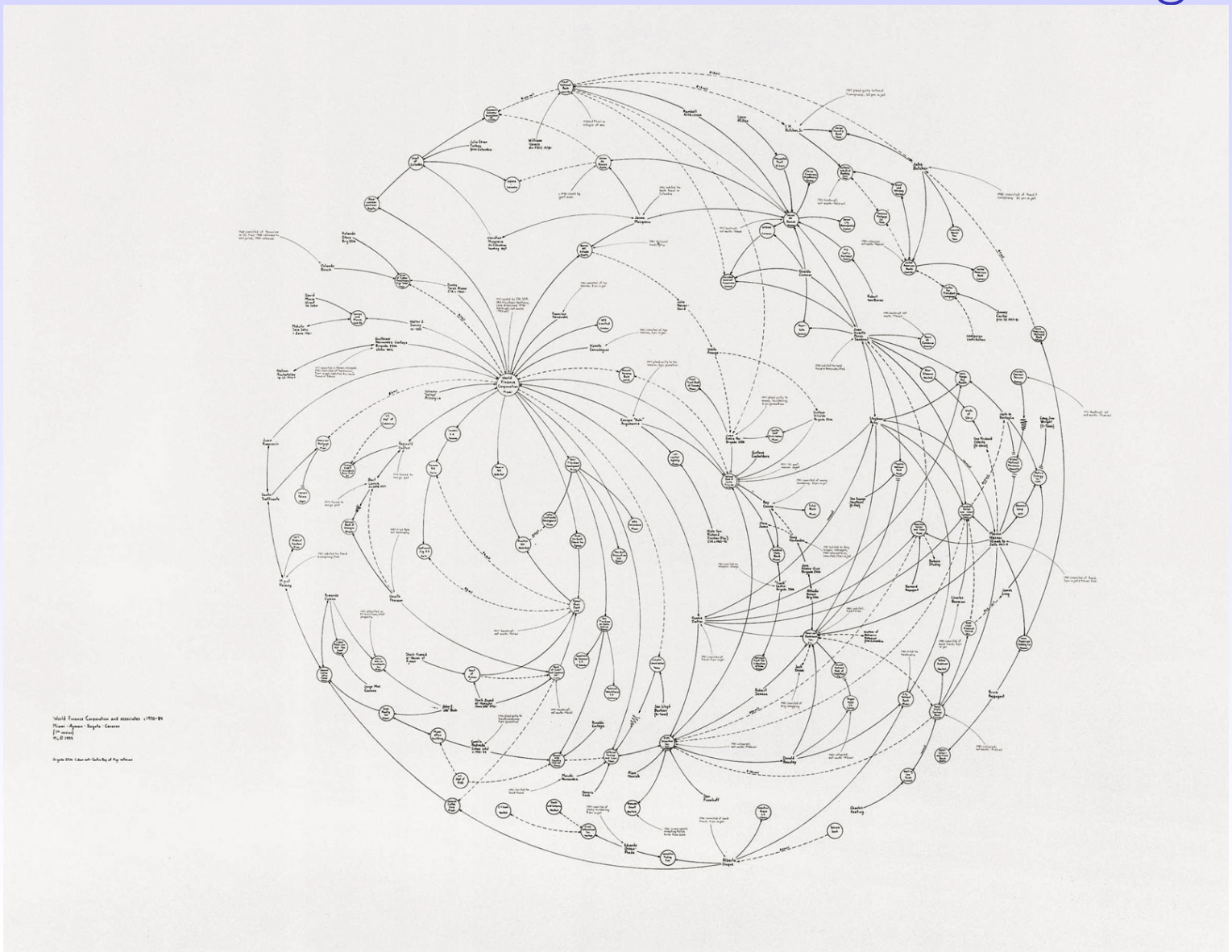
Fabien de Montgolfier
fm@irif.fr

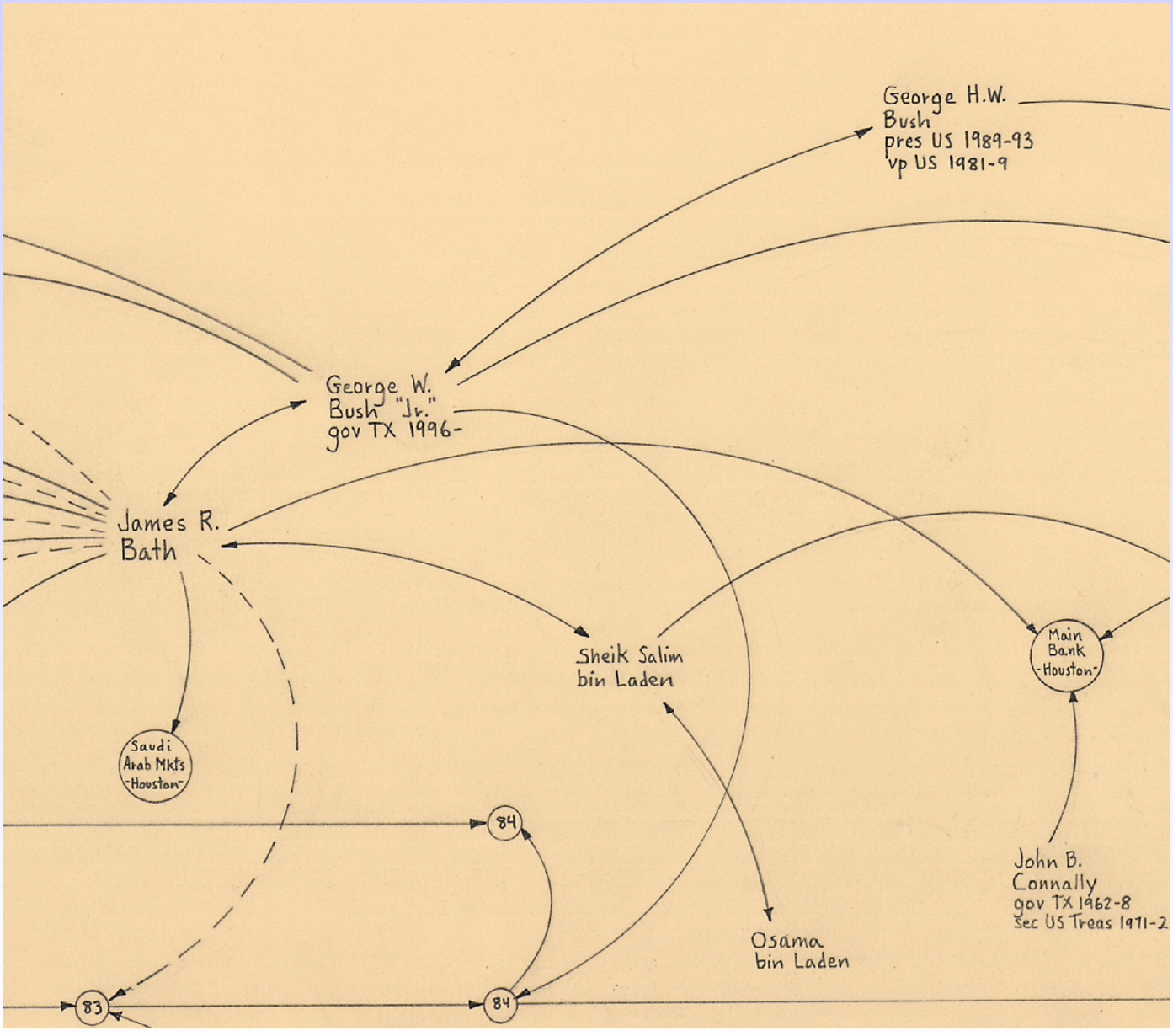
2 décembre 2022

Propagation du Covid à Hong Kong [Adam et al., Nature 2020]



Mark Lombardi «*Global Networks*» artiste d'investigation




George H.W.
Bush
pres US 1989-93
vp US 1981-9
George W.
Bush "Jr."
gov TX 1996-
James R.
Bath
Sheik Salim
bin Laden
Main
Bank
-Houston-
Saudi
Arab Mkts
-Houston-
84
83
84
Osama
bin Laden
John B.
Connally
gov TX 1962-8
Sec US Treas 1971-2

THE
OPTE PROJECT

home about the internet prints / licences faq lgl

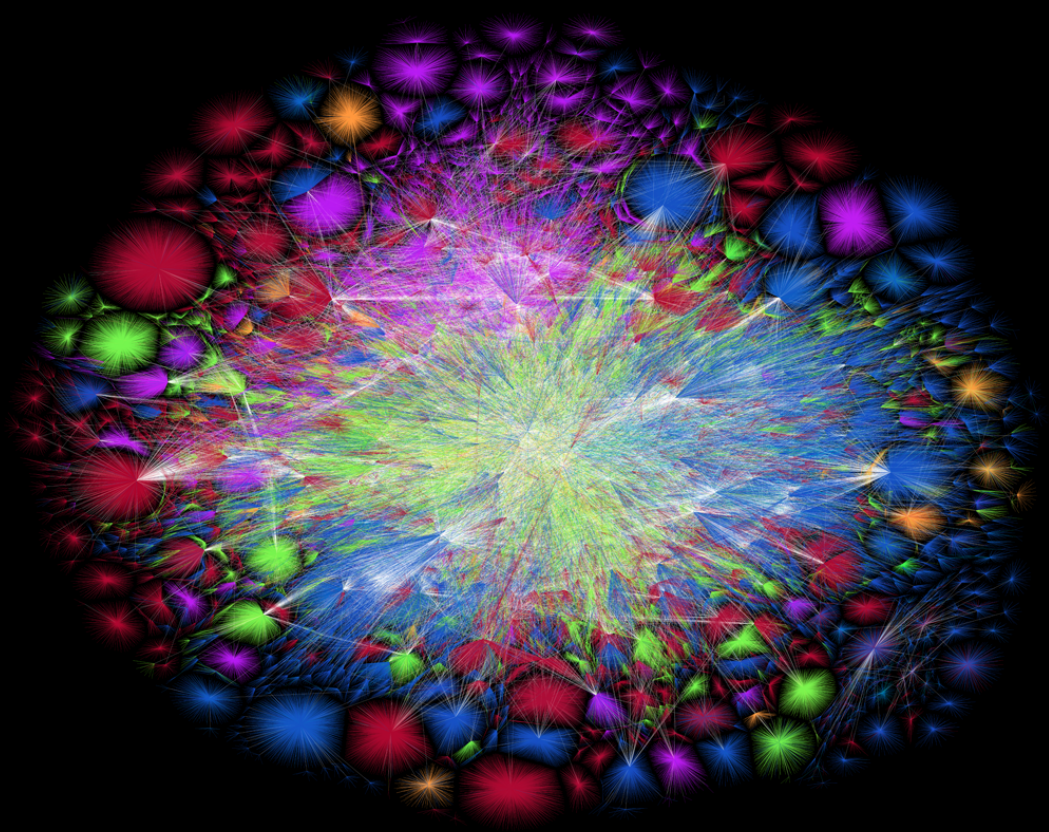
THE INTERNET 2015

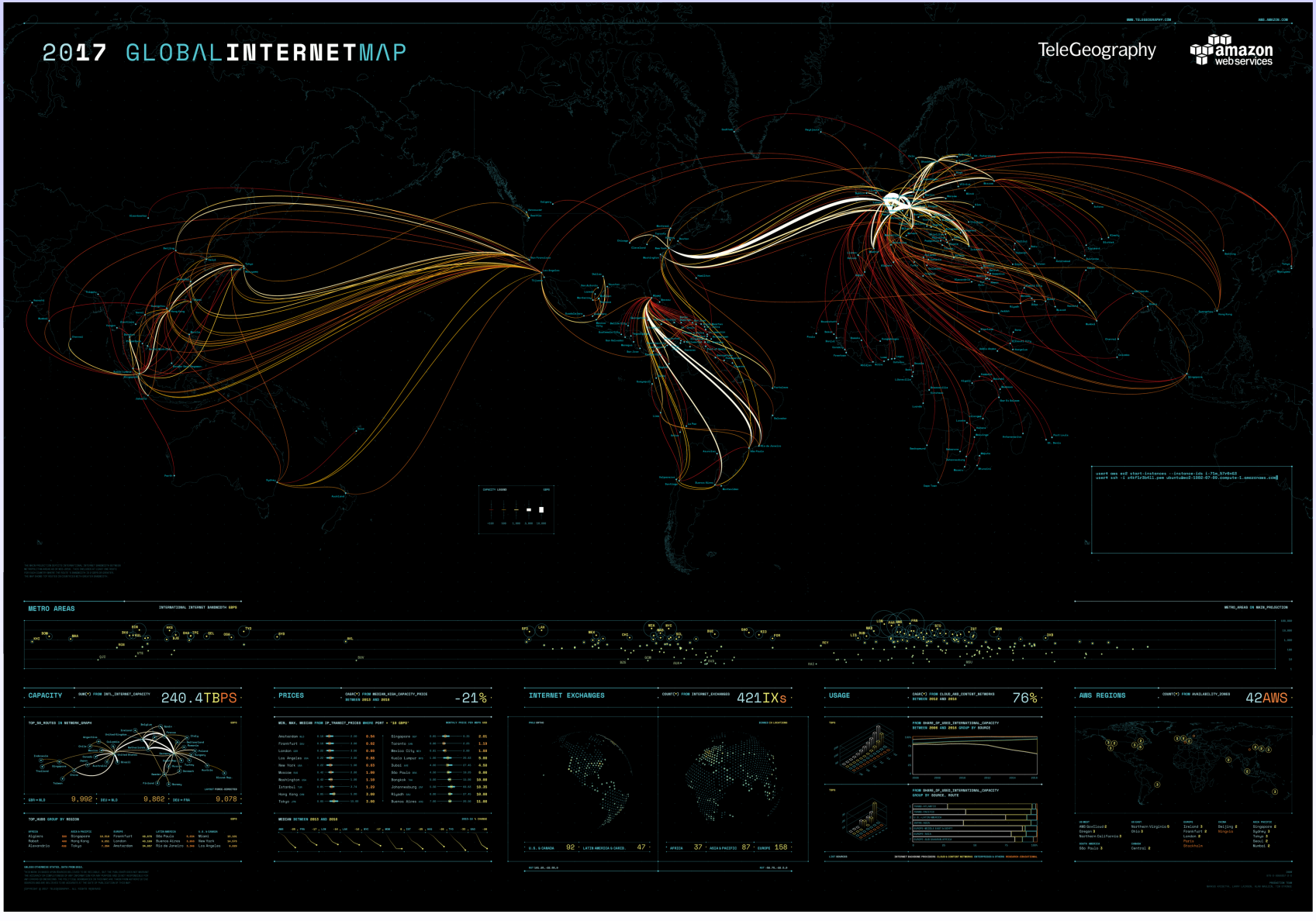
This is the first major release of the Internet map since 2010.

Color Chart:

- North America (ARIN)
- Europe (RIPE)
- Latin America (LACNIC)
- Asia Pacific (APNIC)
- Africa (AFRINIC)
- "Backbone" (highly connected networks)

Date: July 11 2015
Graph Engine: LGL 1000×800 px (png) 10000×8000 px (non-antialiased) (png)




2017 GLOBALINTERNETMAP
TeleGeography
amazon web services
METRO AREAS
CAPACITY
240.4TBPS
PRICES
-21%
INTERNET EXCHANGES
421IXs
USAGE
76%
AWS REGIONS
42AWS

Organisation du cours

- Séances : 2h cours suivies, 2h de TP une semaine sur 2 (deux groupes)
- Note : 50% examen terminal, 50% TPs à rendre
- un TP = un programme a rendre sur Moodle. Banque d'exemples pour validation.

But du cours

- Comprendre le monde !
  - **paramètres** pour décrire les propriétés structurelles (distances, connectivité)
  - **modèles** (aléatoires) pour distinguer ce qui est attendu de ce qui est surprenant
- Savoir **implémenter** des algorithmes et utiliser des outils d'analyse sur de **très grands** graphes (millions de sommets).

Plan du cours

1. **Description**
   Comprendre les propriétés structurelles des GRI
   en informatique, sociologie, biologie, physique, linguistique...
   - **en TP** : Calcul de divers paramètres.
     Big data $\rightarrow$ structures de données et algorithmes efficaces
2. **Modélisation**
   Classifier ces réseaux d'interaction.
   Modèles aléatoires.
   Si même propriétés que le réel, alors même génération ?
   - **en TP** : Génération aléatoire
3. **Deux applications**
   Dissémination (contagion)
   P2P (DHT ; diffusion temps réel)

Static Analysis 2022-2023

Aim

Build correct tools to find bugs in code

```
1.  public class AClass {
2.  public static void main(String args[]) {
3.      int x = 1;
4.      int y = 3/(x++ - 1);
5.  }}
```

Aim

Build correct tools to find bugs in code

```
[ ...]$ analyse AClass.java
AClass.java:3:  error:  Divide By Zero
Expression '0' could be zero at line 3.

1.  public class AClass {
2.  public static void main(String args[]) {
3.  >    int x = 1;
4.       int y = 3/(x++ - 1);
5.  }}

Found 1 issue

  Issue Type(ISSUED_TYPE_ID): #
  Divide By Zero(DIVIDE_BY_ZERO): 1
```

Aim

Build sound languages to write bugless code

```
let _ =
"foo" + 42;;
```

Aim

Build sound languages to write bugless code

```
[ ...]$ ocaml aprog.ml
let _ =
"foo" + 42;;
  Line 2, characters 2-7:
  2 | "foo" > 42;;
      ^^^^^
 Error:  This expression has type string
 but an expression was expected of type int
```

Prerequisites

Know either OCaml or Haskell

Static Analysis 2022-2023

Foundations order theory, results on fixed-points

Dataflow Analysis

- WHILE$_{NNH}$
- Heart of optimizing compilers
- Objective: implement an analysis

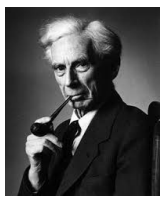
Abstract Interpretation

- WHILE$_{RY}$
- INFER, ABSINT, MOPSA, FRAMA-C, ... written in OCAML
- Objective: implement an abstract interpreter

Along with historical remarks





Static Analysis 2022-2023

https://fbinfer.com/

Final grade

100 % project

- implementation of monotone frameworks
  - generic framework
  - instance to perform constant propagation
- implementation of abstract interpretation

Programmation Orientée Objets : Concepts Avancées

- POCA - deuxième édition.
- Seulement pour ceux qui n'y étaient pas inscrits à la première période.
- Responsable : Mathieu Gandin et Bilal Abbad

Objectifs

- Améliorer votre maîtrise de la Programmation Orientée Objet en l'appliquant dans un cadre proche du monde professionnel
- Approche par la pratique
- Prioriser ce qui va vous servir souvent.

Organisation

- **Premier cours : lundi 9 janvier**
- Avant le premier cours : installez `typescript`, parcourir *https://www.typescriptlang.org/docs/handbook/intro.html*
- `https://moodle.u-paris.fr/enrol/index.php?id=10762`
- Un devoir hebdomadaire
- 1h de cours, avec corrections du devoir et discussion
- Puis suivi par petits groupes
- Évaluation en continu

Administration Système et Réseaux

Figure: Birth of Unix: D. Ritchie, K. Thompson & PDP11 (∼1970)

Administration Système et Réseaux

From: RMS%MIT-OZ@mit-eddie (Richard Stallman)
Newsgroups: net.unix-wizards,net.usoft
Subject: new Unix implementation
Date: Tue, 27-Sep-83 12:35:59 EST

Free Unix!

Starting this Thanksgiving I am going to write a complete Unix-compatible software system called GNU (for Gnu's Not Unix), and give it away free to everyone who can use it. Contributions of time, money, programs and equipment are greatly needed.

Administration Système et Réseaux

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Date: 25 Aug 91 20:57:08 GMT

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

Administration Système et Réseaux

Posted on Fr 30 April 2010
Lennart Poettering

Rethinking PID 1

If you are well connected or good at reading between the lines you might already know what this blog post is about. But even then you may find this story interesting. So grab a cup of coffee, sit down, and read what's coming.

This blog story is long, so even though I can only recommend reading the long story, here's the one sentence summary: we are experimenting with a new init system and it is fun.

"Any sufficiently advanced technology is indistinguishable from magic."

Arthur C. Clarke

Administration Système et Réseaux

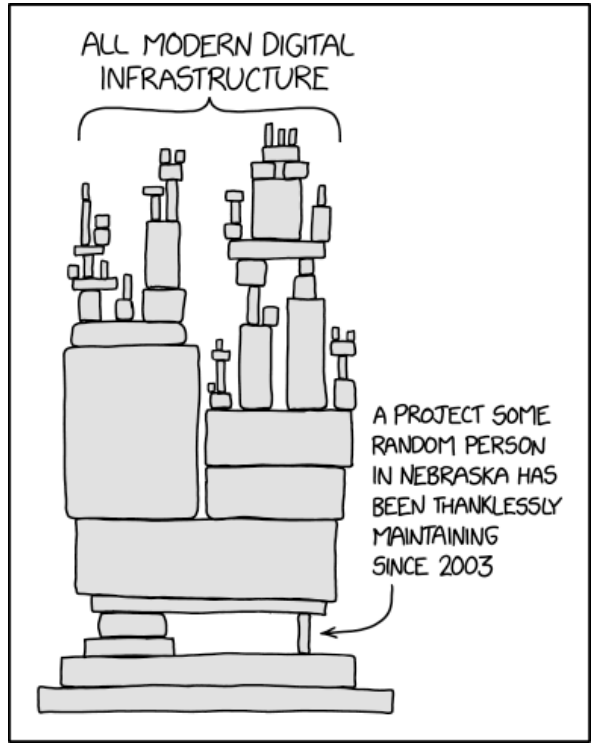


Figure: `xkcd` #2347

Administration Système et Réseaux

Objectives

- Craft a system by hand
- Summon a network with the command line
- Ascend to the Cloud

Coursework

- Lab work (not graded)
- Final exam

Présentation du cours
Méthodes formelles : Approche probabiliste

Responsable :
Arnaud Sangnier
sangnier@irif.fr

BUT :
Étudier des méthodes de vérification automatique pour des modèles probabilistes de systèmes

- Rappel sur la logique temporelle linéaire (LTL) et la vérification
- Étude de différents modèles probabilistes (chaines de Markov à temps discret, processus de décision markovien)
- Étude de propriétés probabilistes
- Étude d'algorithmes de vérification automatique
- Utilisation de l'outil de vérification PRISM

Fonctionnement

- Séance de 3h – la plupart du temps une partie cours et une partie Td/Tp
- Il faut installer Prism sur sa machine et venir avec sa machine
- Il y a un DM sur Prism (30%) et un examen de fin d'année (70%)

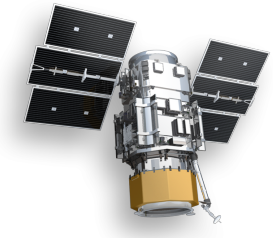
Prérequis

- Algorithmique sur les graphes
- Logique (y compris logique du premier ordre avec quantificateur)
- Automates
- Avoir suivi les cours de Modélisation et Spécification et/ou Méthodes formelles de Vérification est un plus
- Avoir un peu d'attrait pour l'informatique théorique (raisonnement, formalisation, preuve)

Mobilité

Matěj Stehlík

matej@irif.fr



Aperçu du cours

- Nous étudierons des problèmes et algorithmes liés à la mobilité, comme par exemple :
  - le calcul d'itinéraire
  - l'optimisation du trafic dans un réseau
  - l'allocation de fréquences
  - les algorithmes *online*
- Il s'agit d'un cours *théorique*, qui s'appuie beaucoup sur la notion des *graphes* et parfois aussi sur la théorie des jeux.
- Le but est de donner des bases théoriques des différents algorithmes, mais il y aura aussi du travail sur l'ordinateur pour s'approprier ces concepts, et pour implémenter les algorithmes.

Vous apprendrez. . .

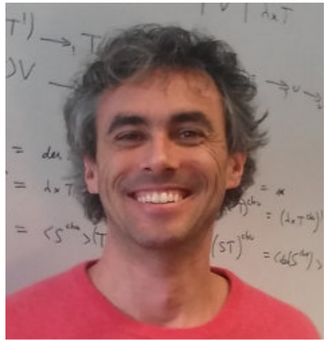
- que trouver le meilleur itinéraire est (beaucoup) plus complexe pour les livreurs que pour les postiers.
- de créer un planning de vols pour une flotte d'avions.
- d'attribuer des fréquences aux antennes d'un réseau cellulaire.
- que (paradoxalement) fermer une route peut améliorer le trafic dans une ville !

Informations complémentaires

- Pré-requis théoriques : graphes (on fera un rappel des notions de base lors du premier cours)
- Pré-requis pratiques : Python ou Java
- Apportez vos propres ordinateurs.
- L'évaluation sera basée sur une épreuve écrite et éventuellement un projet/devoir maison.

Introduction à l'apprentissage profond

(en PyTorch)



Michele Pagani (CM+TP)

Daniela Petrisan (TP)

présentation cours M2 2022/2023

Pb programmation → Pb optimisation

= ○○○○○○○○○○○○○○•••○○○○○

input
28x28 = 784 float

model

learning parameters

label
7 ≃ ?

○○○○○○○○○○○○○○○○○○○○

output
10 float

loss

$$\text{argmin}_p \left(\sum_{(\texttt{input},\texttt{label})} \textbf{loss}(\textbf{model}(\texttt{input}, p), \texttt{label}) \right)$$

Apprentissage profond

=

input

28x28 = 784 float

learning parameters

model

output

10 float

```
##################
## DEFINE MODEL ##
##################
class LeNet5(nn.Module):
    """
    Input - 1x32x32
    Output - 10
    """
    def __init__(self):
        super(LeNet5, self).__init__()
        self.conv1 = nn.Conv2d(1, 6, kernel_size=(5, 5))
        self.conv2 = nn.Conv2d(6, 16, kernel_size=(5, 5))
        self.conv3 = nn.Conv2d(16, 120, kernel_size=(5, 5))
        self.fc1 = nn.Linear(120, 84)
        self.fc2 = nn.Linear(84, 10)

    def forward(self, input):
        layer1 = F.relu(self.conv1(input))                          # 28x28x6
        layer2 = F.max_pool2d(layer1, kernel_size=(2, 2), stride=2) # 14x14x6
        layer3 = F.relu(self.conv2(layer2))                         # 10x10x16
        layer4 = F.max_pool2d(layer3, kernel_size=(2, 2), stride=2) # 5x5x16
        layer5 = F.relu(self.conv3(layer4))                         # 1x1x120
        layer6 = F.relu(self.fc1(torch.flatten(layer5,1)))          # 84
        layer7 = self.fc2(layer6)                                   # 10
        return layer7
```

```
##############################
## TRAINING AS MINI-BATCH GD ##
##############################
def train_loop(train_loader, model, loss_function, epochs):

    # Transfers model to GPU
    if torch.cuda.is_available():
        device = 'cuda'
        model.cuda()
    else:
        device = 'cpu'

    print(device)
    # Train model
    for epoch in range(epochs):
        for images, labels in train_loader:
            # Transfers data to GPU
            images, labels = images.to(device), labels.to(device)
            # Primal computation
            output = model(images)
            loss = nn.CrossEntropyLoss()(output, labels)
            # Gradient computation
            model.zero_grad()
            loss.backward()
            # Update parameters
            with torch.no_grad():
                for p in model.parameters():
                    p -= 0.02 * p.grad

        # for monitoring
        print(f'Train - Epoch {epoch+1}, Loss: {loss.detach().cpu().item()}')
```

```python
###############
## EXECUTION ##
###############
lenet5 = LeNet5()
print(f"Before learning, accuracy = {validate(test_loader, lenet5.cpu())}%")
train_loop(train_loader, model=lenet5, epochs=12)
print(f"After learning, accuracy = {validate(test_loader, lenet5.cpu())}%")
```

```
pagani@odette:~$ /usr/bin/python3.9 /home/pagani/deeplearning/presenation/lenet5.py
Before learning, accuracy = 9%
cuda
Train - Epoch 1, Loss: 2.2595834732055664
Train - Epoch 2, Loss: 0.5144934058189392
Train - Epoch 3, Loss: 0.3164558708667755
Train - Epoch 4, Loss: 0.4379177391529083
Train - Epoch 5, Loss: 0.3646761178970337
Train - Epoch 6, Loss: 0.364197582006454 47
Train - Epoch 7, Loss: 0.153398394584655 76
Train - Epoch 8, Loss: 0.095523208379745 48
Train - Epoch 9, Loss: 0.196038946509361 27
Train - Epoch 10, Loss: 0.086288534104824 07
Train - Epoch 11, Loss: 0.051805865019559 86
Train - Epoch 12, Loss: 0.035702634602785 11
After learning, accuracy = 97%
pagani@odette:~$
```

Sujets centraux

- Rappel des principes de l'apprentissage automatique
- Les tenseurs et leur manipulation
- Réseaux de neurones à propagation avant
- Le processus d'apprentissage
- Les réseaux de convolution
- Les techniques de régularisation

Plus de détails sur:

http://www.informatique.univ-paris-diderot.fr/playground/cours/m2/iap10

Pré-requis et évaluation

- Pré-requis:
  - maîtrise programmation en Python

- Des plus, mais pas nécessaire:
  - bases d'algèbre linéaire, statistiques
  - cours de fouille de données et aide à la decision

- Évaluation:
  - 50% projet + 50% examen final

Programmation Logique et Par Contraintes Avancée

Ralf Treinen

November 29, 2022

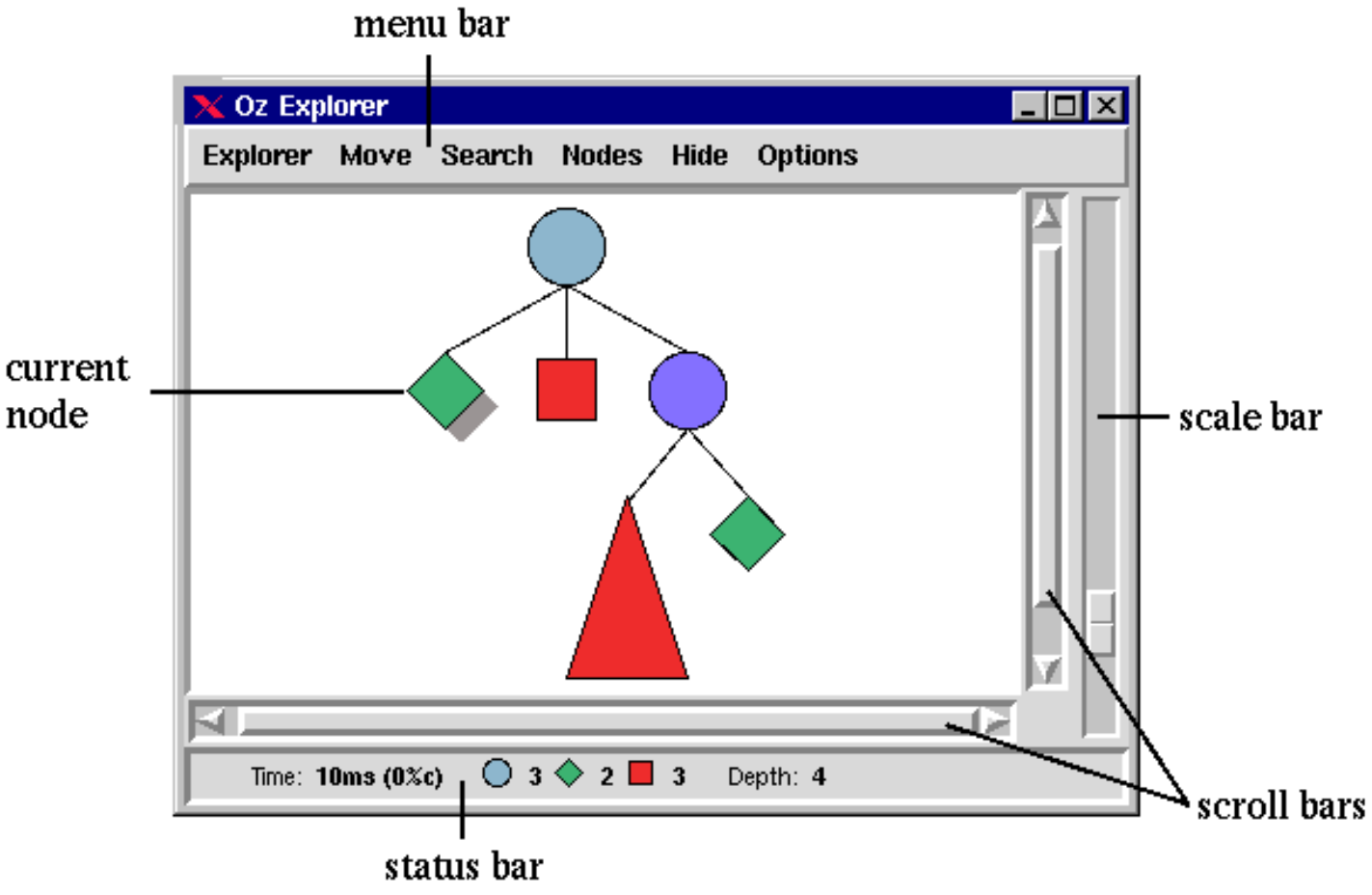
Programmation par contraintes : pourquoi ?

- Résoudre des problèmes qui demandent la *recherche* d'une solution
- Résolution de problèmes combinatoires:
  - Problèmes de planification
  - Problèmes d'ordonnancement optimal
  - Problèmes de placement optimal sous contraintes
  - Résolution des dépendances entre paquet Linux
  - Des puzzles ...

Programmation par contraintes, est-ce du Prolog ?

- Non !
- Nous allons programmer en *Oz* qui est un vrai langage de programmation.
- Oz permet de faire la programmation fonctionnelle, impérative, orienté objet, des interfaces graphiques, de la distribution ...
- Construction d'un arbre de recherche peut être seulement une composante d'un programme complexe : la recherche est *encapsulée*.
- Mécanisme de calcul original, pas de Résolution.

Outil graphique pour interagir avec un arbre de recherche



Pre-requis

- On auriez besoin de :
  - Logique : logique propositionnelle et les bases de la logique du premier ordre, notion de base de la résolution de contraintes (simplification d'un système d'équations, par exemple)
  - Bases de la programmation fonctionnelle (OCaml, Haskell, Scala, F#, ...)
- Mais on vous n'auriez **pas** besoin de :
  - Des connaissances de Prolog
  - Le cours de Programmation Logique du M1

Organisation du cours

- Format : 1.5h de cours, puis 1.5h de TP.
- Examen final (2h)
- Contrôle continu : TP noté dans la deuxième moitié du semestre
- Note : 50% CC + 50% Examen
- `http://www.irif.fr/~treinen/teaching/plpc/`

Méthodes algorithmiques pour l'accès à l'information numérique

MAAIN

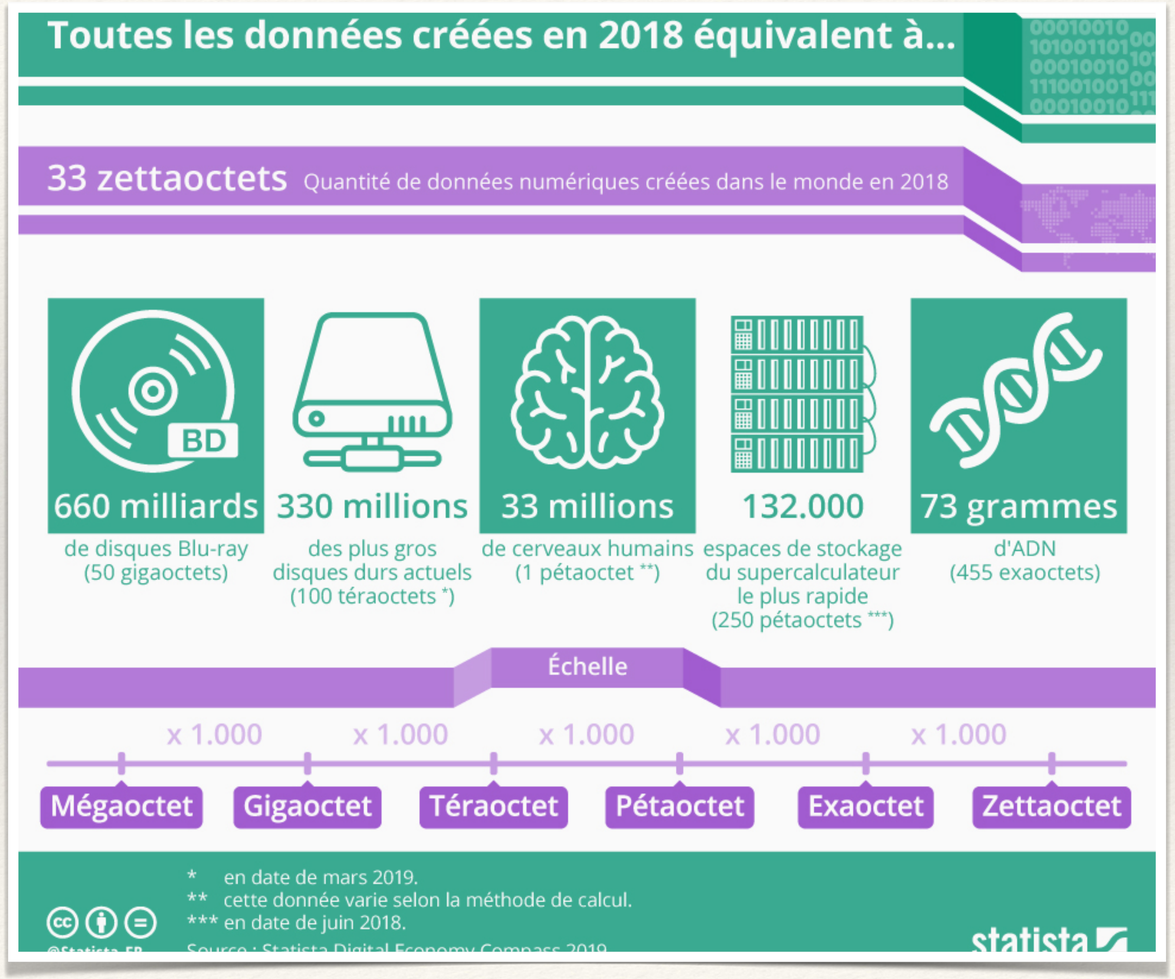
Cours-td & TP : Sylvain Perifel

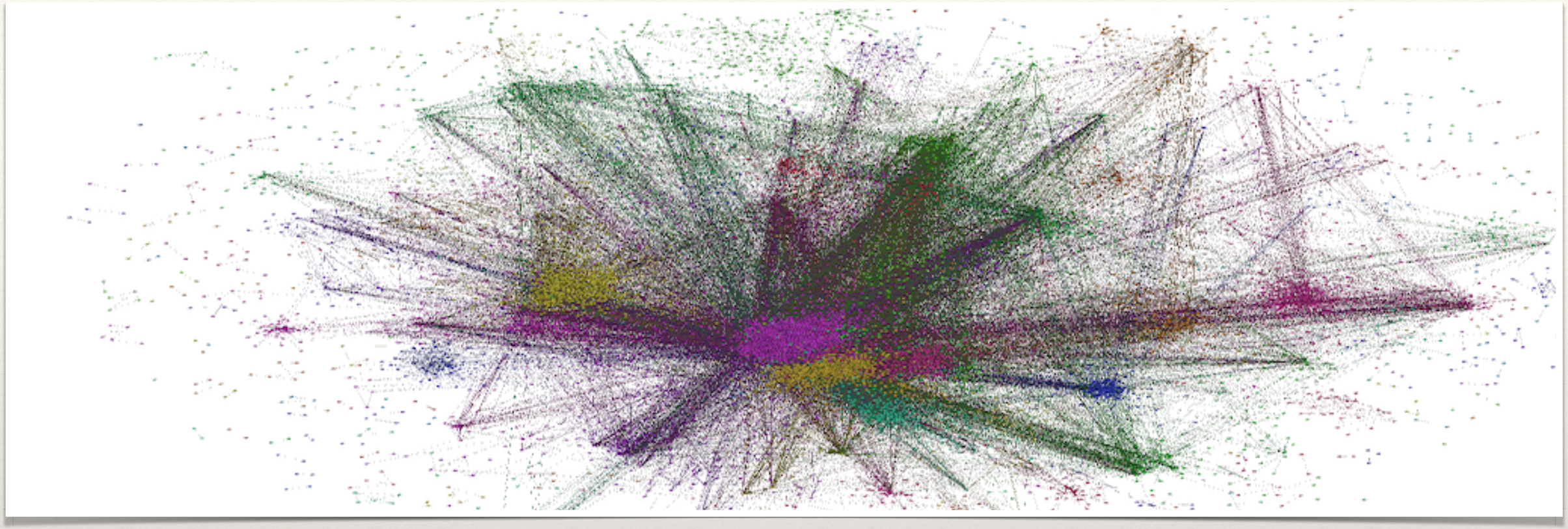
M2 2022-2023

Trouver une aiguille dans une botte de web

- Des milliards de pages sur internet
- Des zettaoctets de données (1 Zo = 10^{12} Go)

Trouver l'information pertinente ???





Graphe du web (Criteo)

1. Moteurs de recherche

Théorie et pratique :

- Aspects algorithmiques
- Programmation

Moteurs de recherche

Algorithmique

Exploration du web

Structuration des données

Calcul de la pertinence des pages (*pagerank*)

Traitement de la requête utilisateur

Projet en binômes en TP

Programmation d'un « **vrai** » **moteur de recherche** sur les pages Wikipédia :

Parser – traitement – pagerank – déploiement

Difficulté gestion mémoire (taille compressée 5 Go) et parsing – langage libre

Le serveur est opérationnel à la fin du cours

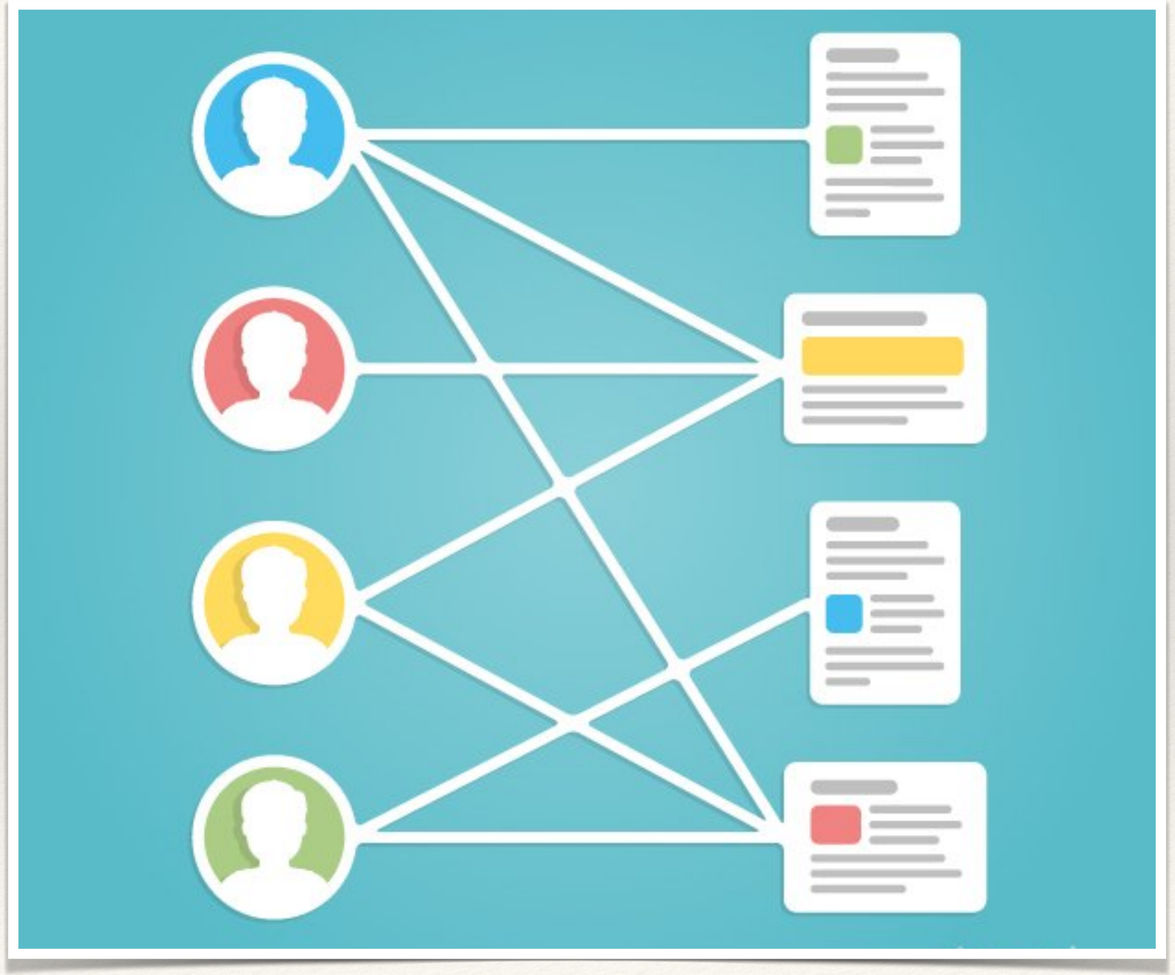
2. Systèmes de recommandation

Recommandations

Comment suggérer la prochaine vidéo pertinente

Comment proposer de nouveaux produits

→ **Algorithmes de recommandation**



Prérequis & organisation

Prérequis :

- un soupçon d'algèbre linéaire
- une once d'algorithmique

1h30 de cours (+ TD)

2h de TP

Évaluation : 50 % examen écrit

50 % projet (TP)

« MAAIN, c'est bien. »

-Un étudiant de M2, 1er avril 2022

Transformation de programmes

Pierre Letouzey

Transformation de programmes, késako ?

- Ancien nom : compilation avancée
- Accent sur les étapes *intermédiaires* d'un compilateur
- Ni parsing ni assembleur x86 (ou en option seulement), mais:
- Remaniement d'AST (arbres de syntaxe)
- Passages progressifs de langages haut-niveau à bas-niveau
- Comment rendre les récursivités terminales ?
- Qu'est-ce qu'une mise en CPS (Continuation Passing Style) ?

Prérequis

Plutôt des recommandations :

- Avoir suivi M1 Compilation est un plus
  - thème commun, même si on verra des techniques différentes
- Connaître au moins un langage fonctionnel
  - On programmera dans le fragment fonctionnel de Scala

Projet

Transformation d'un langage idéalisé mais représentatif (traits impératifs + fonctionnels) vers du bytecode JVM en assurant une récursivité efficace.

Evaluation

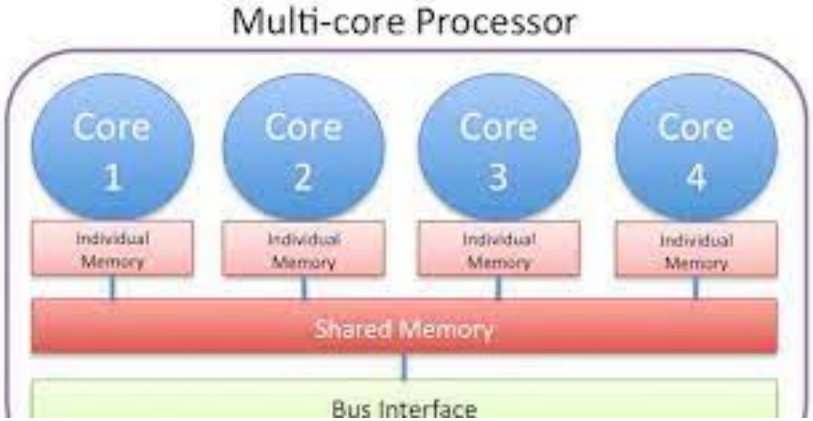
65% projet + 35% examen

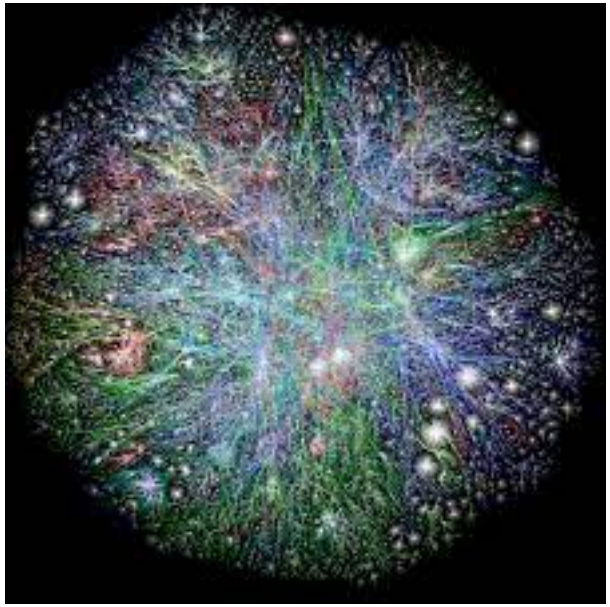
Programmation Comparée

- Ce cours est déjà complèt avec les étudiants qui doivent le suivre obligatoirement.
- Premier cours : **Jeudi 12 janvier**

Algorithmique répartie

Algorithmique répartie





Algorithmique Répartie

Divers aspects de l'algorithmique répartie.

- Du local au global (exemple de la terminaison distribué) et algorithmes sur les réseaux de communication (sans défaillances)
- Défaillances - en synchrone et asynchrone
  - Attaque coordonnée
  - Consensus
  - Généraux byzantins

Algorithmique répartie

Le cours:

- Algorithmique et concept
- 2-3 devoirs + examen final.
- (Suivre aussi -si possible- le cours de programmation répartie)

Architecture des Systèmes d'Information

Emmanuel Fuchs

Programme du cours

- Architecture d'Entreprise
- Infrastructure Matérielle des systèmes d'information
- Infrastructure Réseaux des systèmes d'information
- Infrastructure Logicielle des systèmes d'information
- Architecture SI Orientée Service (SOA)
- Architecture SI Orientée Service WEB
- Architecture SI Orientée Service REST
- Framework d'entreprise
- Framework Logiciel