Master 2 Informatique Fondamentale et Appliquée Année 2021-2022

Amélie Gheerbrant, Arnaud Sangnier, Ralf Treinen

06/12/2021

Calendrier 21/22

- Deuxième période : du 03/01 au 18/03 (cours/TD/TP)
  Examens : 21/03 au 25/03
- Troisième période : du 28/03 au 30/09 (stage)
  Soutenance de stage, et jury du M2 : mid/fin septembre

Vous devrez valider 60 crédits

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

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)

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)

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

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	Grands réseaux d'interaction
Programmation Synchrone	Interfaces et Outils de MacOS-X
Protocoles des services Internet	Programmation répartie
	Systèmes avancés

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 de Bases de Données	Architecture des Systèmes d'Information
Méthodes Formelles de Vérification	Programmation Logique et Par Contraintes Avancée
Modélisation et Spécification	Programmation Répartie
Programmation Synchrone	Interfaces et Outils de MacOS-X
	Typage et Analyse Statique

Inscriptions pédagogiques à des modules

- Vous devrez choisir les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.

Inscriptions pédagogiques à des modules

- Vous devrez choisir les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.
- Vous choisissez 6 cours d'informatique pour la deuxième période.

Inscriptions pédagogiques à des modules

- Vous devrez choisir les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.
- Vous choisissez 6 cours d'informatique pour la deuxième période.
- 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 les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.
- Vous choisissez 6 cours d'informatique pour la deuxième période.
- 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 les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.
- Vous choisissez 6 cours d'informatique pour la deuxième période.
- 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 première période : du 6/12 (après cette réunion) au 10/12.

Inscriptions pédagogiques à des modules

- Vous devrez choisir les cours que vous voulez suivre : inscription pédagogique. Lien sur le wiki de l'UFR.
- Vous choisissez 6 cours d'informatique pour la deuxième période.
- 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 première période : du 6/12 (après cette réunion) au 10/12.
- Les cours ont des capacités limitées. La priorité sera donnée aux étudiants pour qui le cours demandé est recommandé.

Informations et contacts

- Le wiki de l'UFR :
  `http://www.informatique.univ-paris-diderot.fr`

Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment SG, `crochet@informatique.univ-paris-diderot.fr`

Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment SG, `crochet@informatique.univ-paris-diderot.fr`
- Responsables de parcours : prendre rendez-vous par email

Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment SG, `crochet@informatique.univ-paris-diderot.fr`
- Responsables de parcours : prendre rendez-vous par email
  - DATA : Amélie Gheerbrant `Amelie.Gheerbrant@irif.fr`

Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment 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`

Informations et contacts

- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment 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`

Informations et contacts

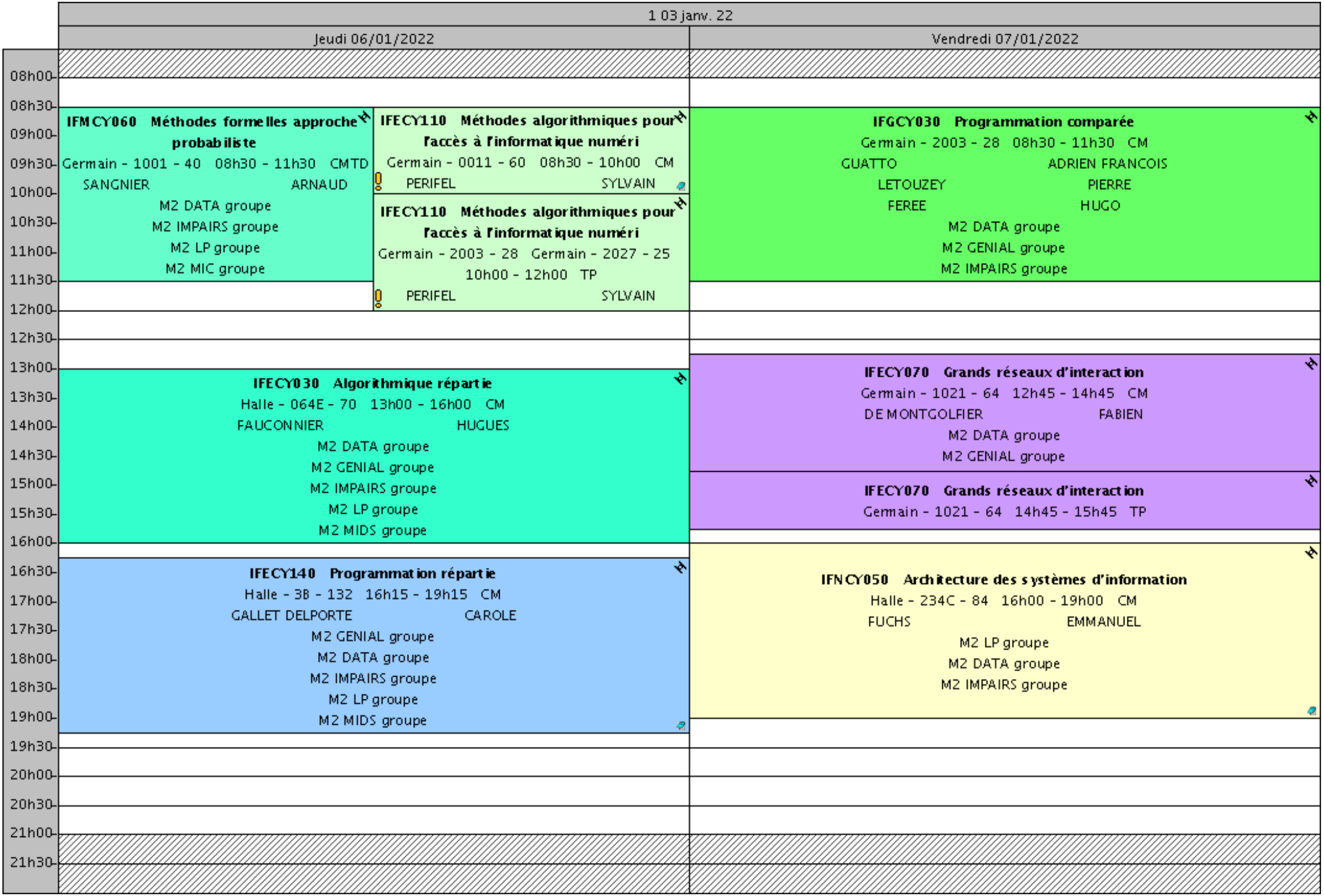
- Le wiki de l'UFR : `http://www.informatique.univ-paris-diderot.fr`
- Suivi administratif : Sylvia CROCHET, bureau 3002, batiment SG, `crochet@informatique.univ-paris-diderot.fr`
- Responsables de parcours : prendre rendez-vous par email
  - DATA : Amélie Gheerbrant `Amelie.Gheerbrant@irif.fr`
  - IMPAIRS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

M2-Lu-Ma-Mer

1 03 janv. 22

	Lundi 03/01/2022	Mardi 04/01/2022	Mercredi 05/01/2022
08h00			
08h30	**IFGCY130 Typage et analyse statique** Germain - 2003 - 28 08h30 - 10h00 CM CHARGÉ DE COURS M2 LP groupe M2 IMPAIRS groupe M2 DATA groupe	**IFECY170 Systèmes avancés** Germain - 2027 - 25 08h30 - 10h30 CM YUNES JEAN BAPTISTE M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe	
09h00			**IFNCY090 Optimisation** Germain - 1001 - 40 09h00 - 10h30 CM VIGER FABIEN M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe
09h30		**IFGCY040 Transformation de programmes** Germain - 2003 - 28 08h30 - 11h30 CM LETOUZEY PIERRE M2 LP groupe M2 IMPAIRS groupe M2 DATA groupe	
10h00	**IFGCY130 Typage et analyse statique** Germain - 2003 - 28 10h00 - 11h30 TP CHARGÉ DE COURS M2 LP groupe M2 IMPAIRS groupe M2 DATA groupe		
10h30		**IFECY170 Systèmes avancés** Germain - 2027 - 25 10h30 - 12h00 TP YUNES JEAN BAPTISTE M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe	**IFNCY090 Optimisation** Germain - 2001 - 27 10h30 - 12h00 TP VIGER FABIEN M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe
11h00			
11h30			
12h00			
12h30			
13h00	**IFECY020 Administration Système et Réseaux** Germain - 1001 - 40 12h30 - 15h30 CM CHARGÉ DE COURS M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe		**IFGCY110 Programmation Logique et par contraintes avancée** Germain - 1001 - 40 13h00 - 14h30 CM TREINEN RALF M2 MIC groupe M2 DATA groupe M2 IMPAIRS groupe
13h30		**IFECY100 Interfaces et outils de MacOs -X** Germain - 2027 - 25 13h15 - 15h15 CM YUNES JEAN BAPTISTE M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe	
14h00			
14h30			**IFGCY110 Programmation Logique et par contraintes avancée** Germain - 2032 - 23 14h30 - 16h00 TP TREINEN RALF M2 MIC groupe M2 DATA groupe M2 IMPAIRS groupe
15h00			
15h30		**IFECY100 Interfaces et outils de MacOs -X** Germain - 2027 - 25 15h15 - 17h15 TP YUNES JEAN BAPTISTE M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe	
16h00	**IFECY120 Mobilité** Halle - 027C - 70 15h30 - 18h30 CM STEHLIK MATEJ M2 DATA groupe M2 IMPAIRS groupe M2 LP groupe		
16h30			
17h00			**IFNCY280 Introduction à l'apprentissage profond** Germain - 2032 - 23 16h00 - 19h00 CMTD PAGANI MICHELE ANGELO M2 DATA groupe M2 LP groupe
17h30			
18h00			
18h30			
19h00			
19h30			
20h00			
20h30			
21h00			
21h30			

M2-Je-Ve


1 03 janv. 22
Jeudi 06/01/2022
Vendredi 07/01/2022
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
IFMCY060 Méthodes formelles approche probabiliste
Germain - 1001 - 40 08h30 - 11h30 CMTD
SANGNIER ARNAUD
M2 DATA groupe
M2 IMPAIRS groupe
M2 LP groupe
M2 MIC groupe
IFECY110 Méthodes algorithmiques pour l'accès à l'informatique numéri
Germain - 0011 - 60 08h30 - 10h00 CM
PERIFEL SYLVAIN
IFECY110 Méthodes algorithmiques pour l'accès à l'informatique numéri
Germain - 2003 - 28 Germain - 2027 - 25
10h00 - 12h00 TP
PERIFEL SYLVAIN
IFGCY030 Programmation comparée
Germain - 2003 - 28 08h30 - 11h30 CM
GUATTO ADRIEN FRANCOIS
LETOUZEY PIERRE
FEREE HUGO
M2 DATA groupe
M2 GENIAL groupe
M2 IMPAIRS groupe
IFECY030 Algorithmique répartie
Halle - 064E - 70 13h00 - 16h00 CM
FAUCONNIER HUGUES
M2 DATA groupe
M2 GENIAL groupe
M2 IMPAIRS groupe
M2 LP groupe
M2 MIDS groupe
IFECY070 Grands réseaux d'interaction
Germain - 1021 - 64 12h45 - 14h45 CM
DE MONTGOLFIER FABIEN
M2 DATA groupe
M2 GENIAL groupe
IFECY070 Grands réseaux d'interaction
Germain - 1021 - 64 14h45 - 15h45 TP
IFECY140 Programmation répartie
Halle - 38 - 132 16h15 - 19h15 CM
GALLET DELPORTE CAROLE
M2 GENIAL groupe
M2 DATA groupe
M2 IMPAIRS groupe
M2 LP groupe
M2 MIDS groupe
IFNCY050 Architecture des systèmes d'information
Halle - 234C - 84 16h00 - 19h00 CM
FUCHS EMMANUEL
M2 LP groupe
M2 DATA groupe
M2 IMPAIRS groupe

Typage et analyse statique 2021-2022

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

Typage et analyse statique 2021-2022

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

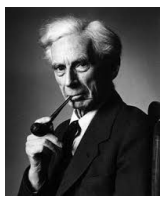
Types

- LAMBDA
- Objective: understand type inference

Along with historical remarks





Typage et analyse statique 2021-2022

https://fbinfer.com/

Final grade

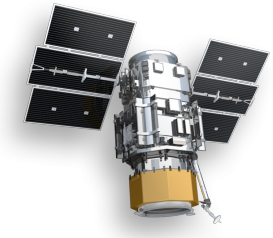
100 % project

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

Mobilité

Matěj Stehlík

matej@irif.fr



Aperçu du cours

- Nous étudions des algorithmes liés à la mobilité, comme par exemple :
  - le routage
  - le calcul d'itinéraires
  - l'auto-organisation. . .
- Objet d'intérêt : réseaux d'entités mobiles capables de communiquer directement (véhicules, robots ou drones, capteurs, etc.)
- Il s'agit d'un cours *théorique*, qui s'appuie beaucoup sur la notion des *graphes*.
- 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.

Informations complémentaires

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

Transformation de programmes

Pierre Letouzey

6 décembre 2021

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

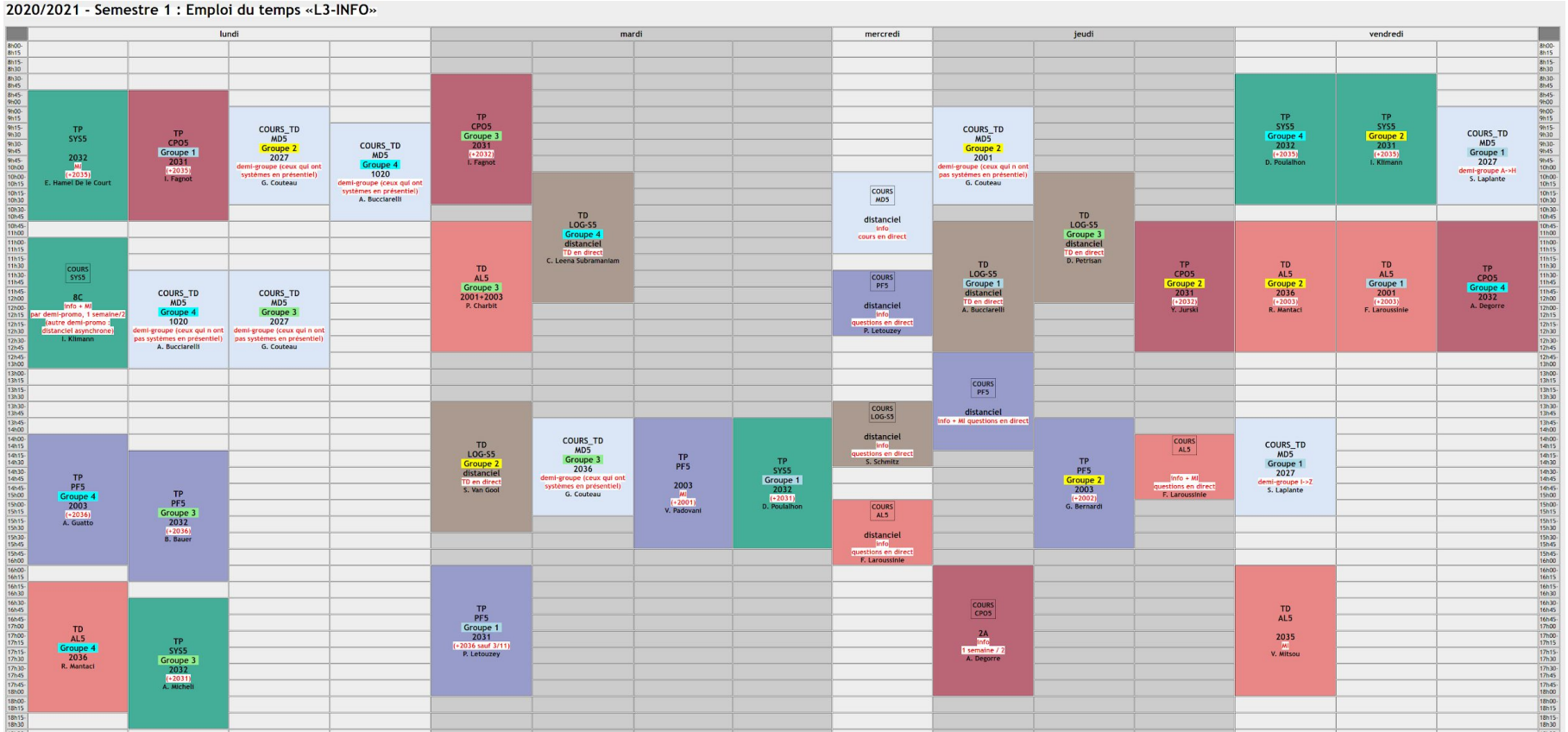
65% projet + 35% examen

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 ?

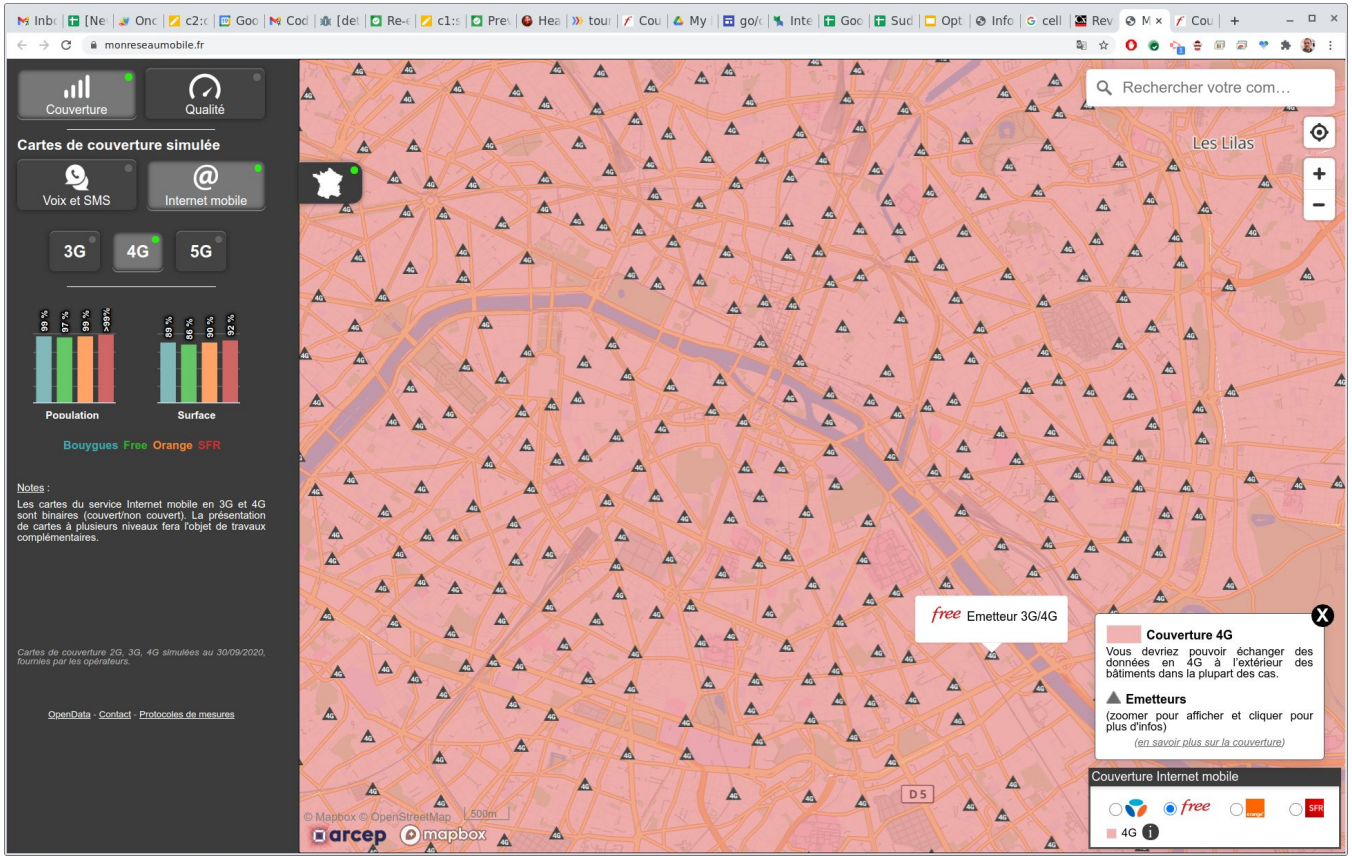
Autre exemple: Sudoku (cliquer sur l’image)

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 9h-12h, Sophie Germain: 1h30 cours + 1h30 TD

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

Cours au tableau (en présentiel)

Pas de support de cours, mais des vidéos (dispo après chaque cours)

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. Arbre couvrant minimum et autres **Algorithmes Gloutons**
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 Logique et Par Contraintes Avancée

Ralf Treinen

December 1, 2021

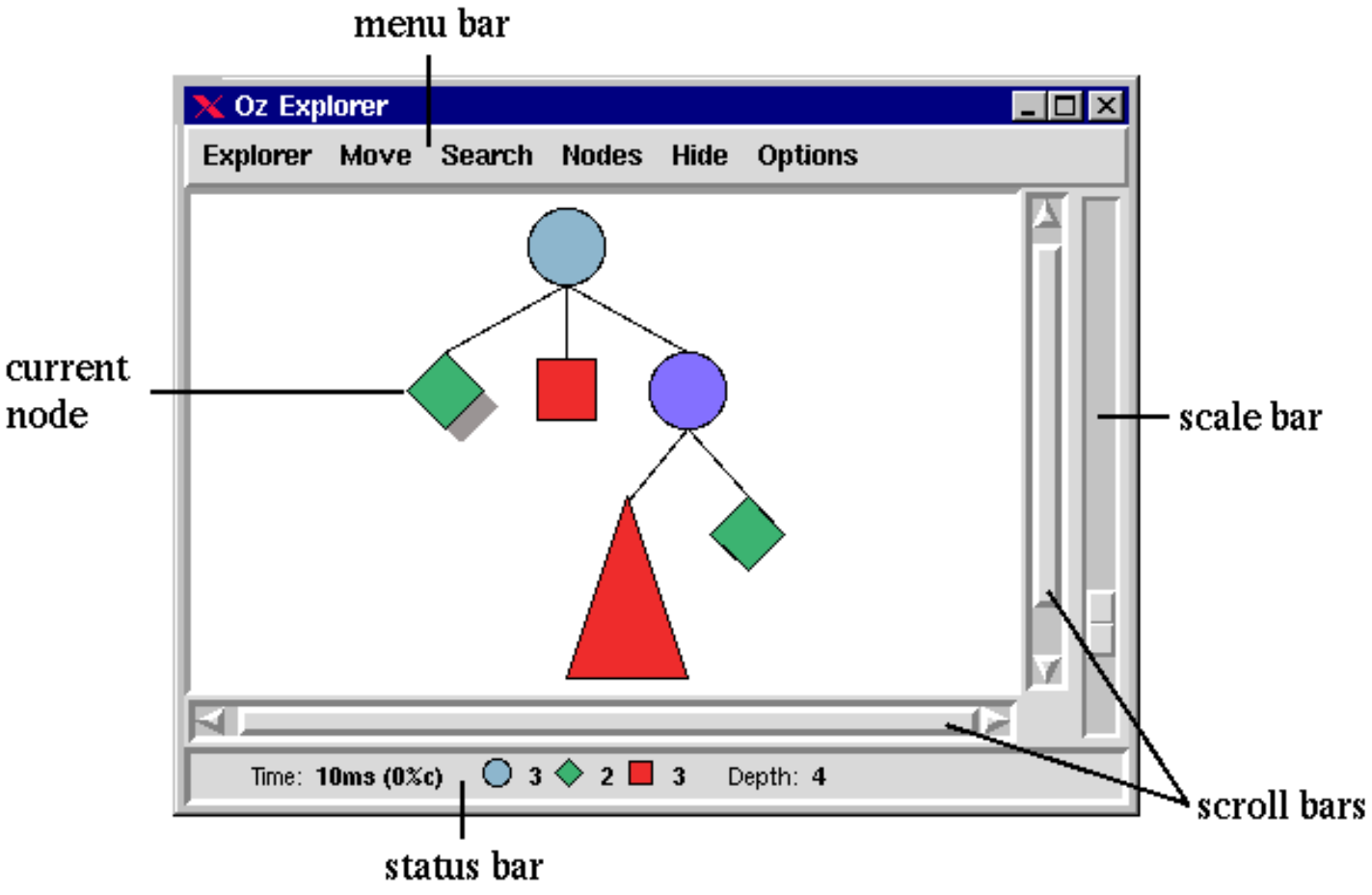
Programmation par contraintes : pourquoi ?

- 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/`

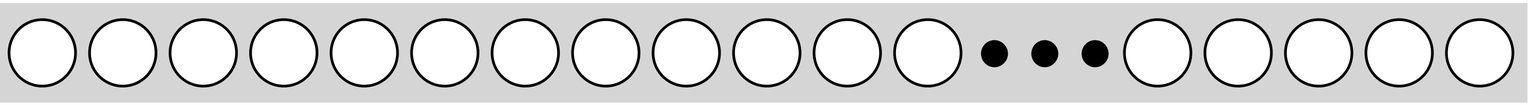
Introduction à l'apprentissage profond

(en PyTorch)

Michele Pagani - présentation cours M2 2021/2022

Pb programmation → Pb optimisation

=

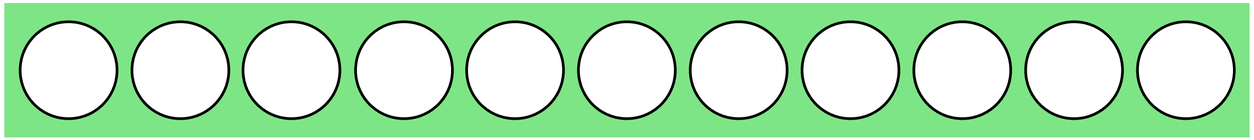


input

28x28 = 784 float

label

7

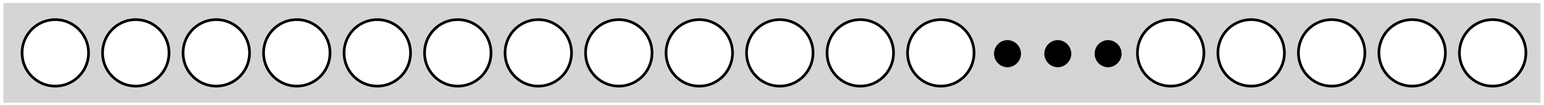


output

10 float

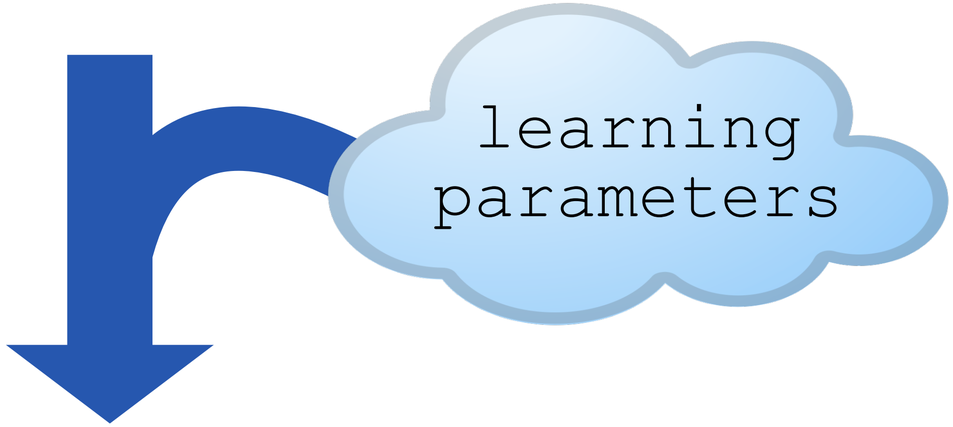
Pb programmation → Pb optimisation

=



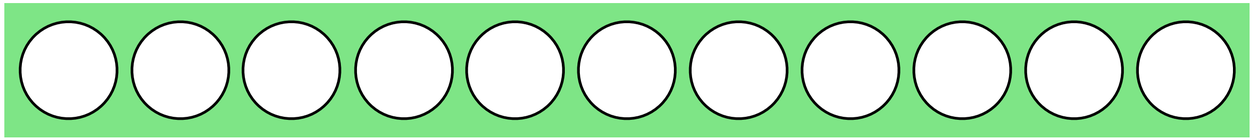
input
28x28 = 784 float

model



label

7



output
10 float

Pb programmation → Pb optimisation

=

input
28x28 = 784 float

model

learning
parameters

label
7 ≃ ?

output
10 float

loss

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:~$
```

Introduction à l'apprentissage profond

(en PyTorch)

Michele Pagani - présentation cours M2 2021/2022

Introduction à l'apprentissage profond

(en PyTorch)

Prérequis :
bonne connaissance de Python
(tranches, objets, itérateurs, context managers, gestion des modules…)

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)

Méthodes algorithmiques pour l'accès à l'information numérique

MAAIN

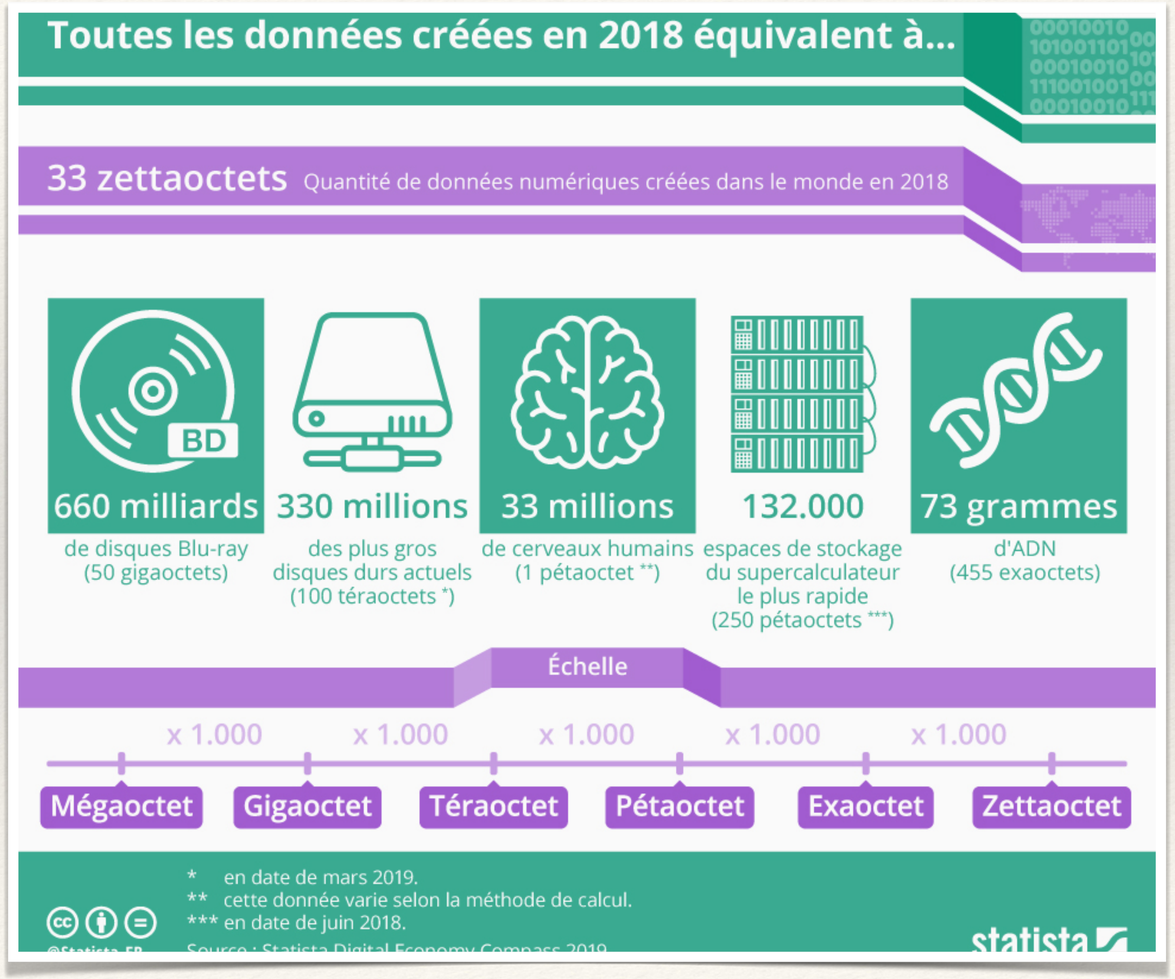
Cours-td & TP : Sylvain Perifel

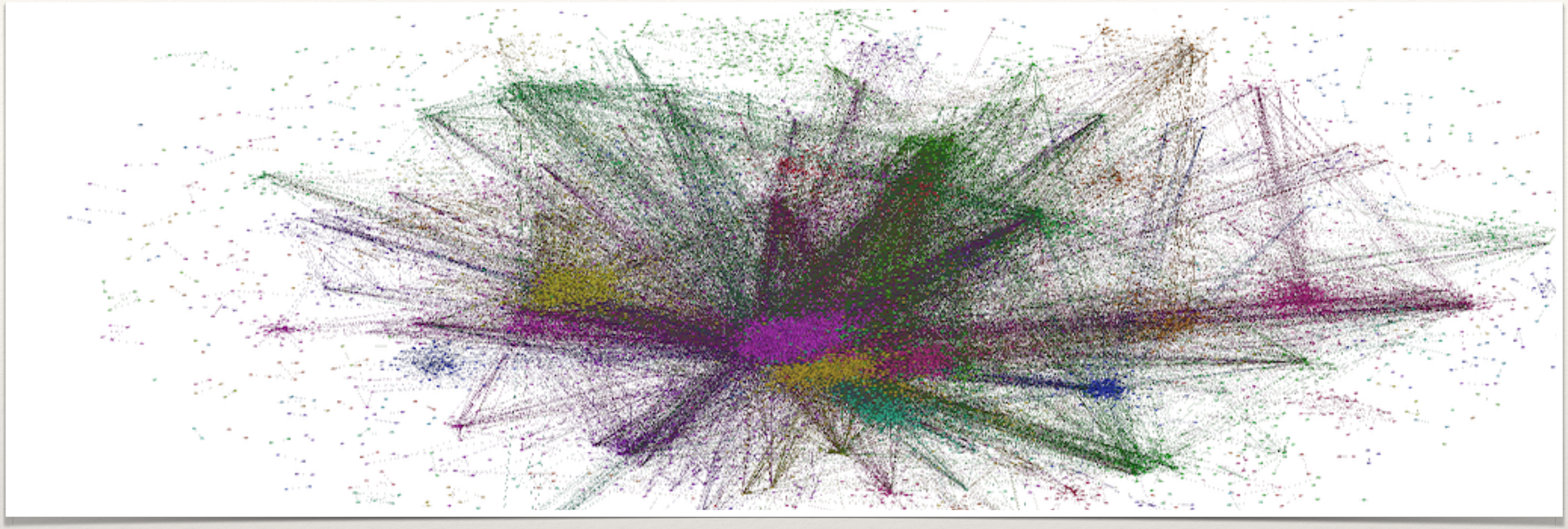
M2 2021-2022

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

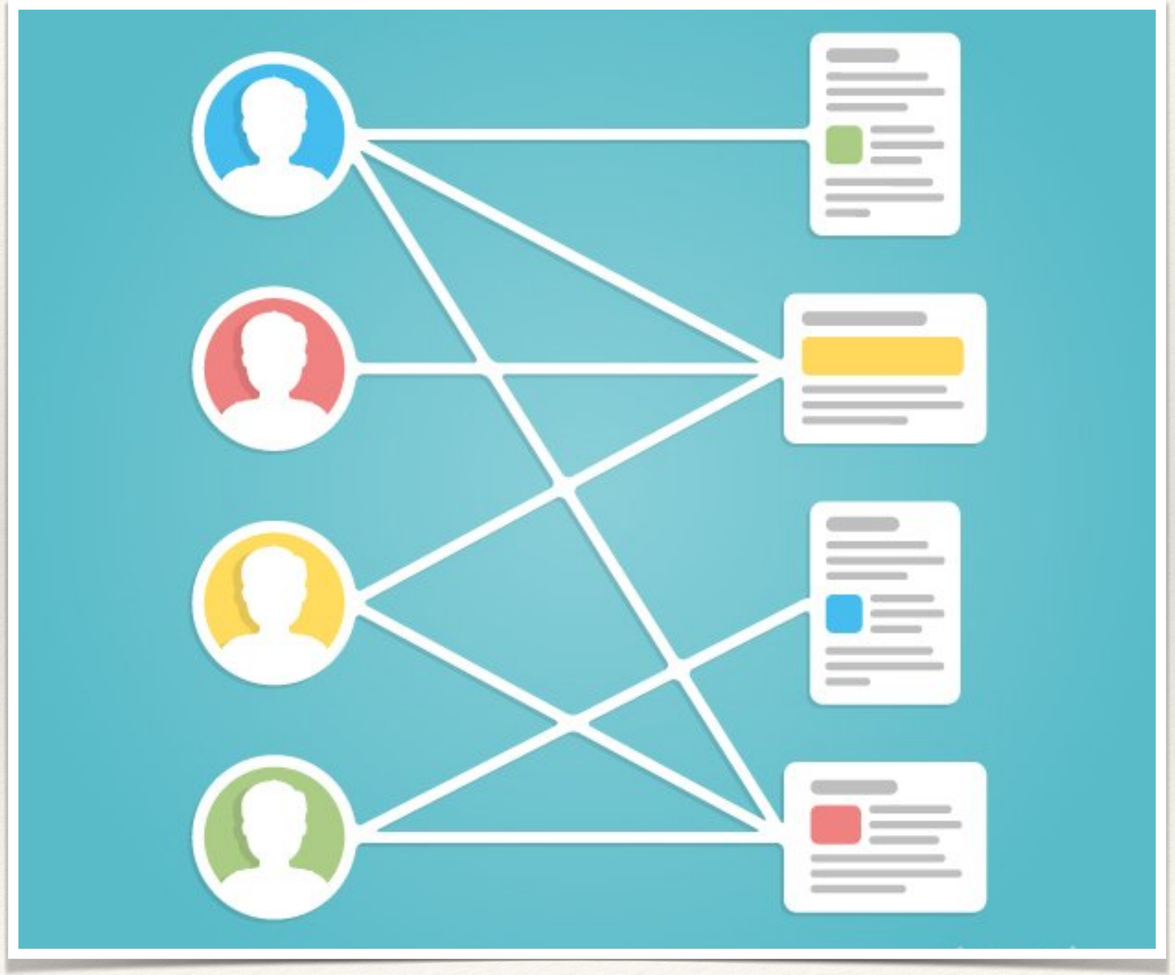
2. Systèmes de recommandation

Recommandations

Comment suggérer la prochaine vidéo pertinente

Comment proposer de nouveaux produits

→ **Algorithmes de recommandation**



Organisation

1h30 de cours (+ TD)

2h de TP

Évaluation : 50 % examen écrit

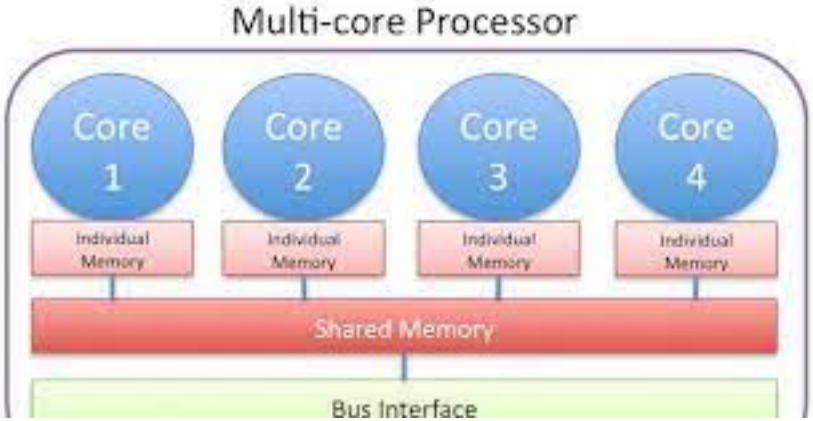
50 % projet (TP)

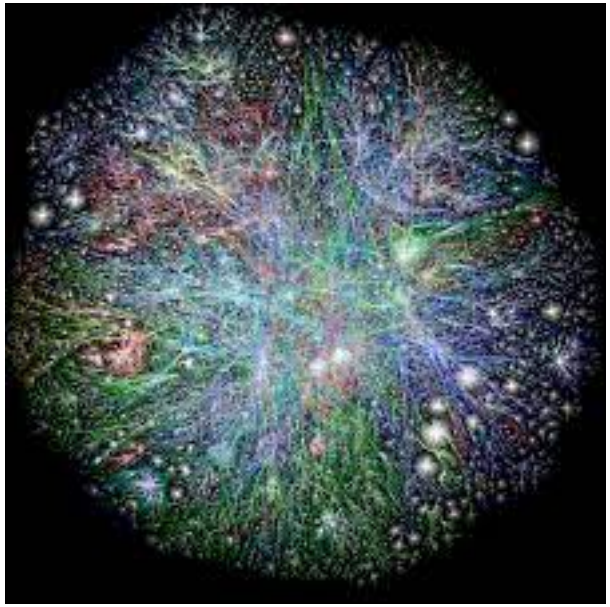
« MAAIN, c'est bien. »

-Un étudiant de M2, 1er avril 2020

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:

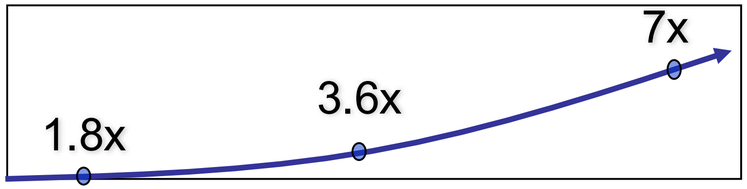
- Algorithmique et concept
- 2-3 devoirs + examen final.
- (Suivre aussi -si possible- le cours de programmation répartie)

Programmation répartie

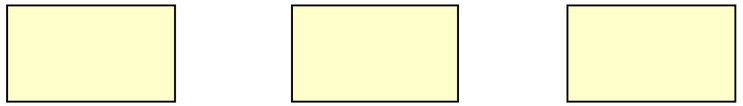
Jeudi 16h15-19h15

Carole Delporte cd@irif.fr

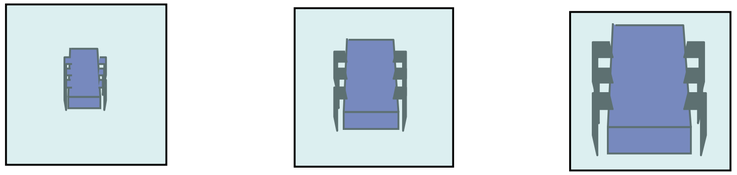
Accélaration



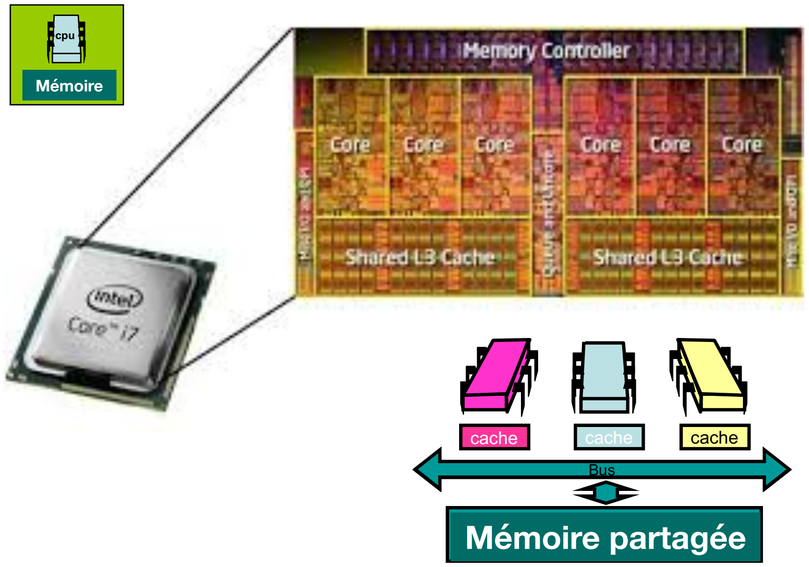
Code utilisateur

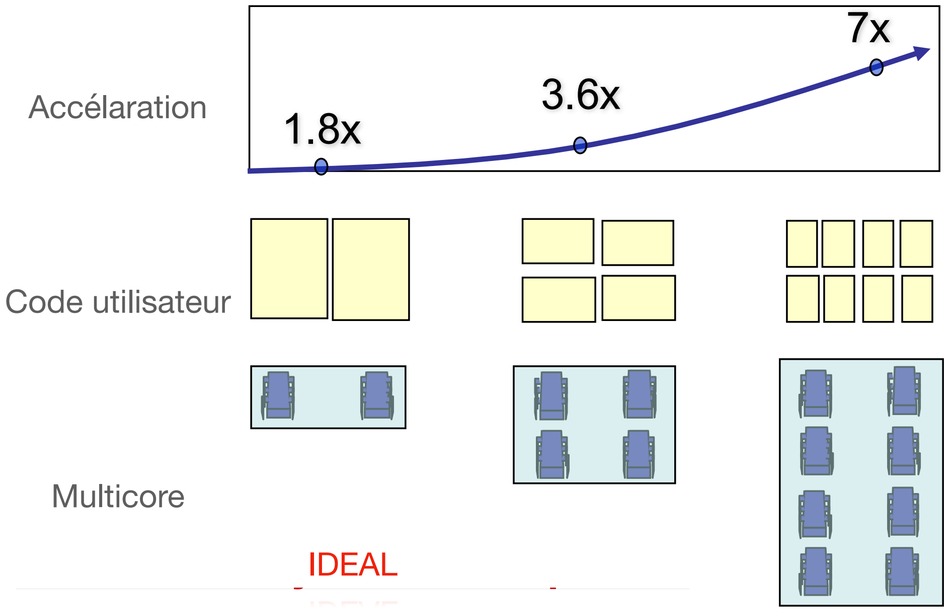


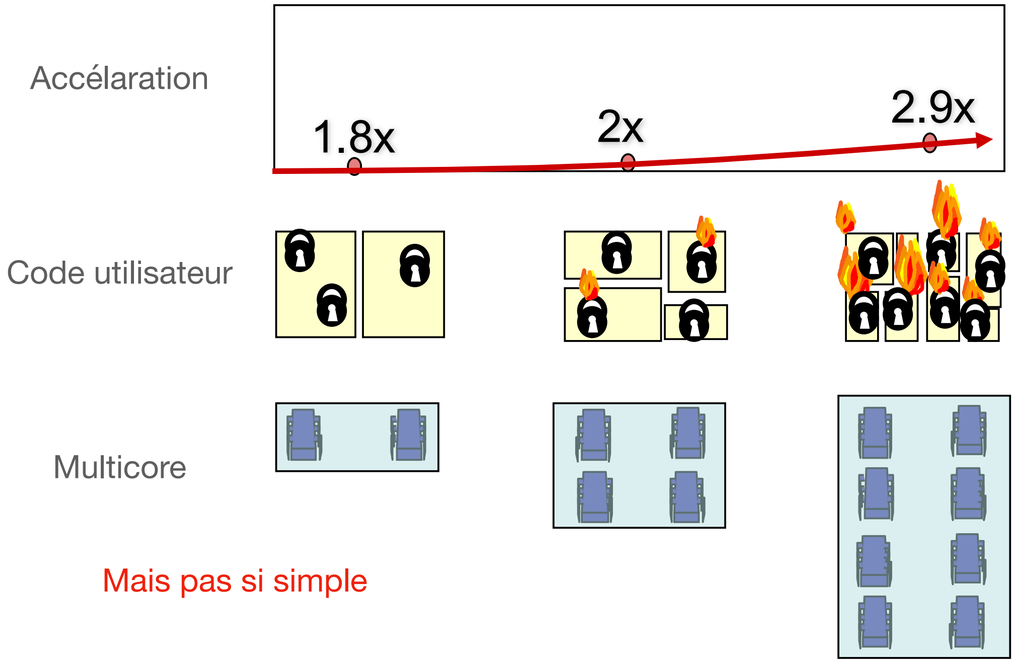
Multicore



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

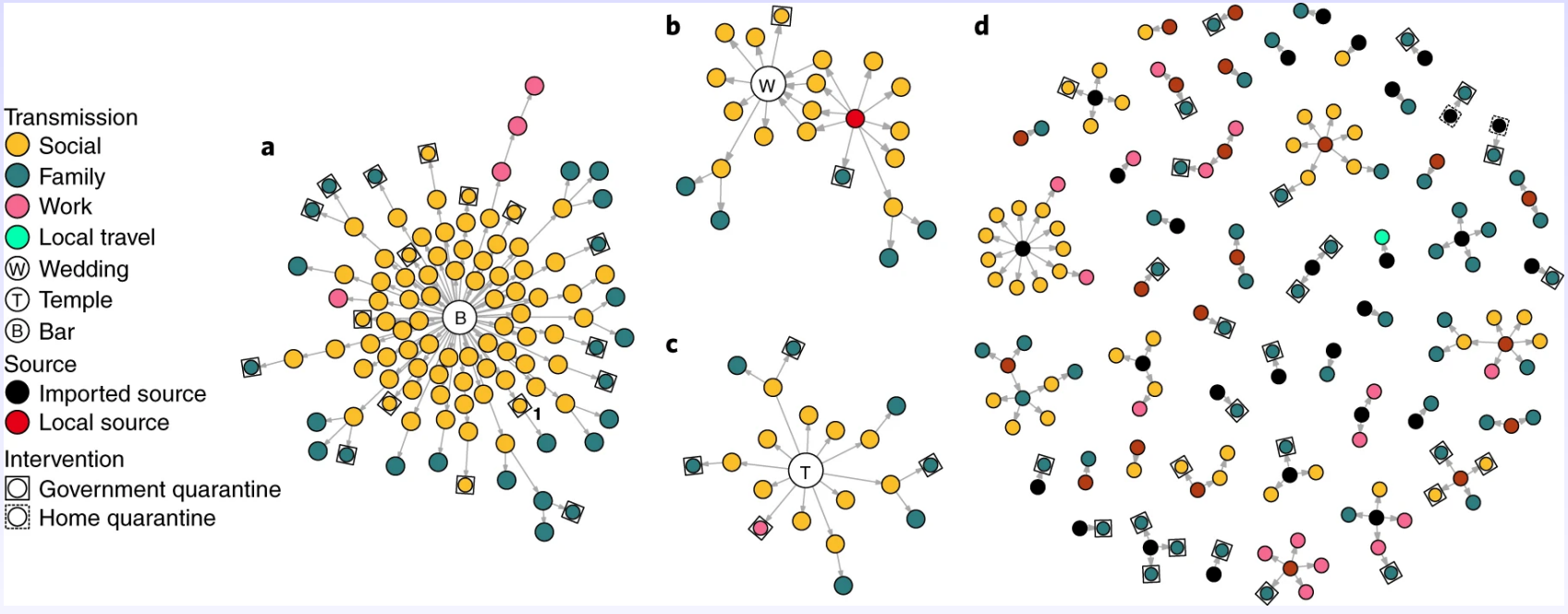
Programmez (en Java) de structures de données concurrentes

Grands Réseaux d'Interaction

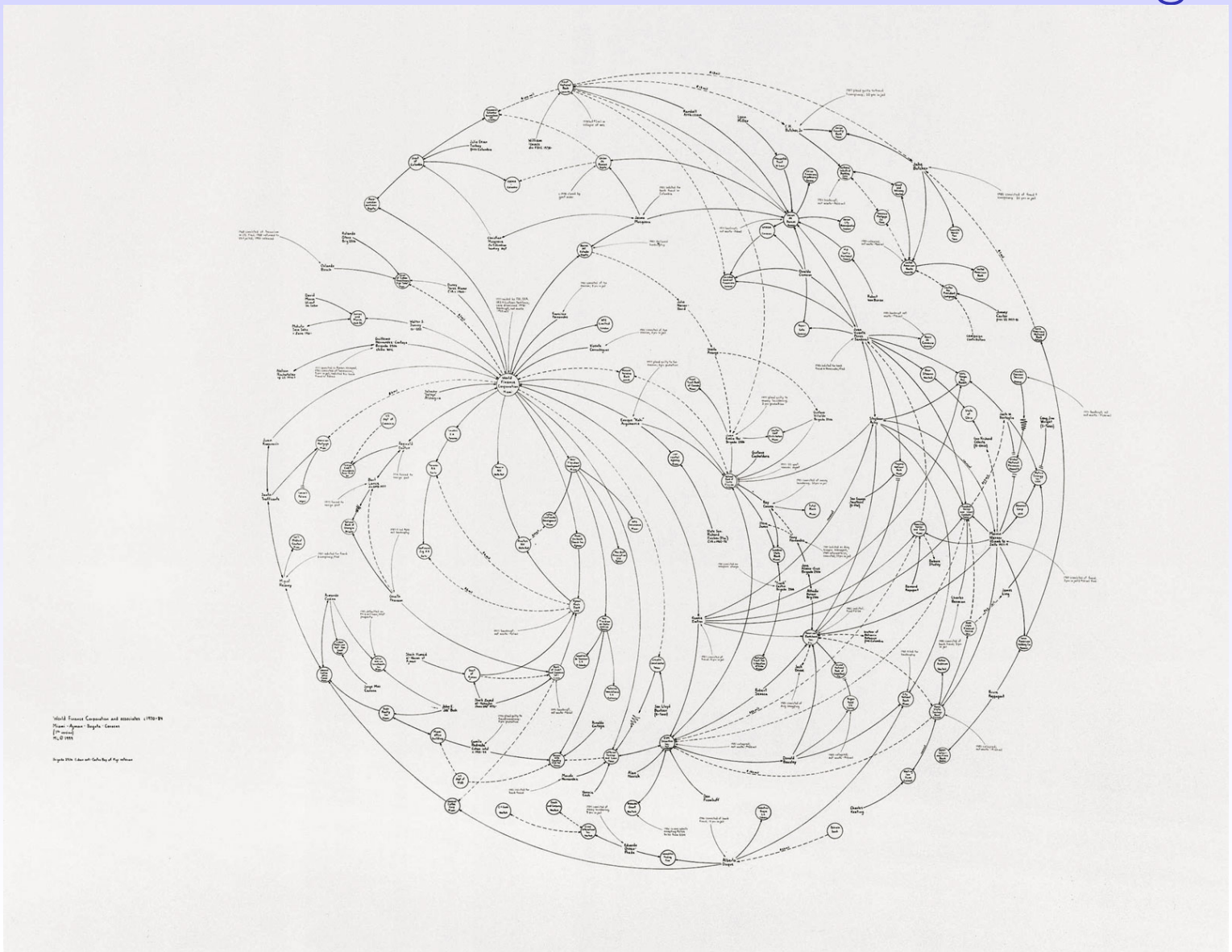
Fabien de Montgolfier
fm@irif.fr

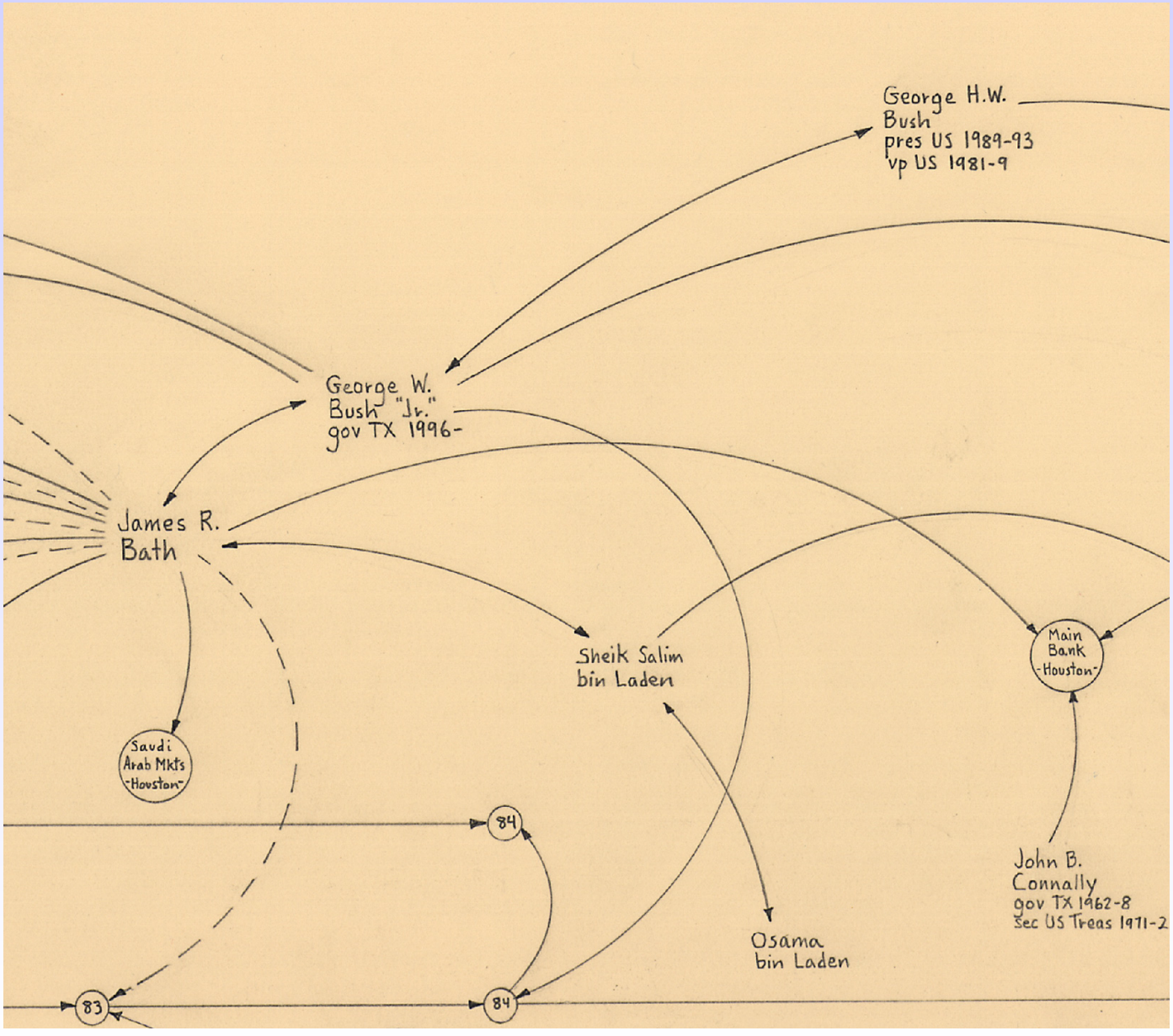
6 décembre 2021

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

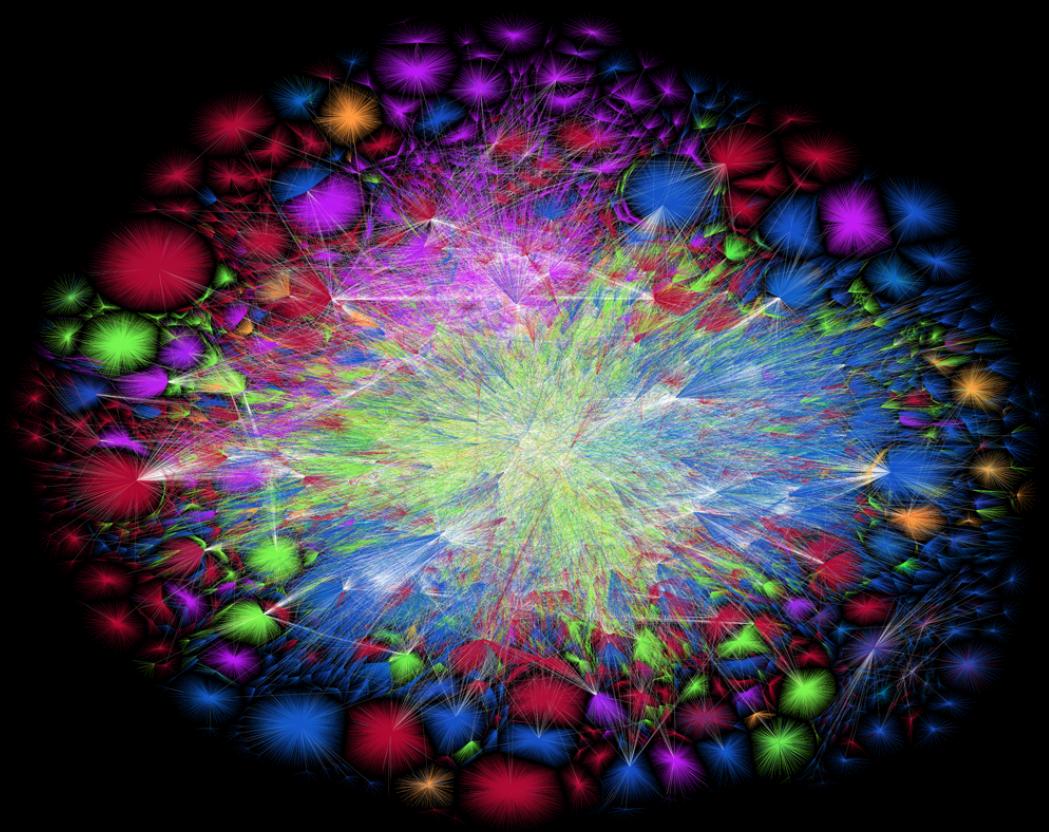
THE INTERNET 2015

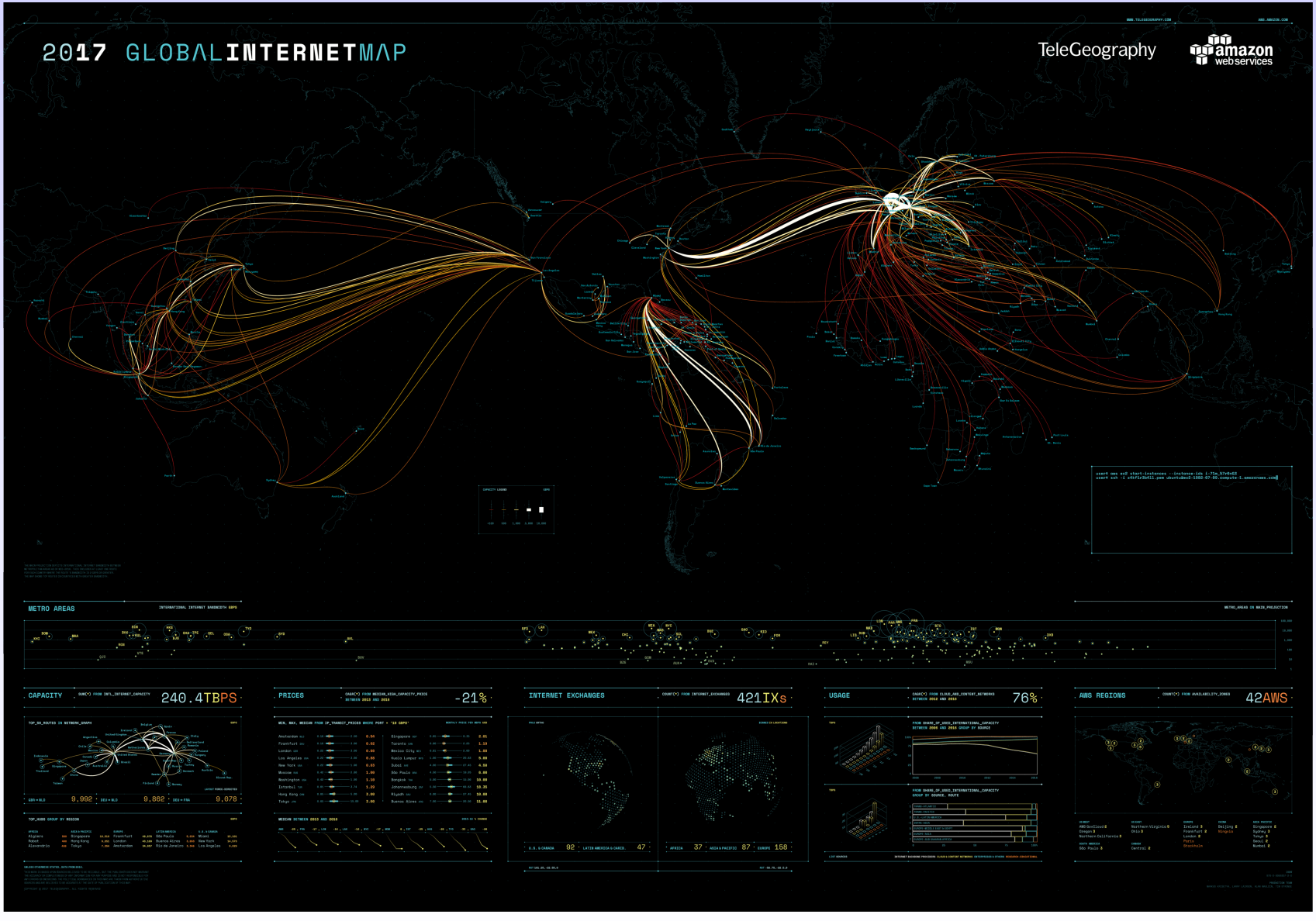
This is the first major release of the Internet map since 2010.

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 par 2h de TP une semaine sur 2
- 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)

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