Liste des leçons d'info pour les oraux d'agrégation (2014)

Par **Lilian Besson**. Toutes remarques sont les bienvenues par courriel ou via bitbucket.

Voir aussi la liste des leçons de maths ? ou la Bible ? ou la liste des leçons d'info en PDF.

Programme

Les leçons se rangent dans 4 sujets :

— 909 910 923 : Automates (finis, à piles) et langages (rec, rat, algé, énumérables etc).

— 912 913 915 915 922 928 : Machines de Turing, calculabilité et décision, classes de complexité.

— 901 902 903 906 921 925 926 927 : Algorithmique : structures de données, tris, graphes, recherche, paradigmes (DVP, glouton, dynamique etc).

— 926 917 918 924 919 920 : Logique : formules, théories, système de preuves.

Il y a **24** leçons d'informatique pour **la session 2014**.

La plupart des liens pointent vers des PDFs dans les dossiers (en accès restreint) :

— a/di/ pour les développements,

— a/li/ pour les plans de leçons.

Fin introduction.

---

901. Structures de données : exemples et applications.

— Structure efficace de dictionnaire, via le hachage parfait, présentation très propre des espaces sur lesquels on fait les calculs probabilistes, comme fait dans le [Cormen, Ch11.5 p238] 901,921, ?

— Tri par tas (en $O(n \log n)$) et explication de la structure de tas binaire (min), complexité de `creerTas` en $O(n \log n)$) [Cormen, ?] et [BBC, Ch3.4 p56] pour les files min et [BBC, Ch5.1.3 p127] pour le tri. On peut aussi proposer l'algorithme de Dijkstra en version optimisée comme application des files de priorités min 901,903,926,(927), ?

Autres idées

— ? Arbres binaires de recherche AVL (recherche, insertion et suppression en $O(\log n)$) (structures efficaces de dictionnaire) [Cormen, en exercice non corrigé] et [BBC, Ch6.2 p148] 901,921, ?

902. Diviser pour régner : exemples et applications.

— Un morceau de l'analyse en moyenne du tri rapide, traité dans [BBC, Ch5.1.1 p122] ou [Aho, Hopcroft, Ullman, Ch8.3 p268] 902,903,926, ?

— Exponentiation rapide : présentation, complexité et preuve (fait dans [BBC, ?], [Cormen, ?]), ou multiplication matricielle rapide par la méthode de Strassen (présentation, complexité, preuve partielle), comme [Cormen, Ch28.2 p710] ou [Dehornoy, Par9.3.4 p214]. Application à l'algorithme de Dantzig pour le calcul de la matrice des plus courts chemins d'un graphe orienté valué (poids $> 0, < 0$ peu importe), qui est en $O(n^3 \log n)$ (un peu mieux que le calcul naïf des itérés successives en $O(n^4)$, mais on ne peut pas descendre en $O(n^{\log 7} \log n)$ car Strassen demande d'avoir un $-$ inverse du $+$ ce qui n'est pas possible sur $\mathbb{R}_{\min} = (\mathbb{R}, \min, +)$, cf [Cormen, Ch25.1.e p607]) 902,(925), ?

Autres idées

— ? Calcul de l'enveloppe convexe de n points du plan par la méthode diviser pour régner, comme fait dans le [Baynat, Ex9.4 p441]. On pourra se référer à [Cormen, Ch33.3 p915] pour un rappel sur les deux autres méthodes (Graham et Jarvis) 902, ?

— ? Sélection en temps linéaire par la médiane des médianes et une bonne récurrence de partition, bien fait dans [Cormen, Ch9.3] mais compliqué à exposer 902, ?

903. Exemples d'algorithmes de tri. Complexité.

— Tri par tas (en $O(n \log n)$) et explication de la structure de tas binaire (min), complexité de `creerTas` en $O(n \log n)$) [Cormen, ?] et [BBC, Ch3.4 p56] pour les files min et [BBC, Ch5.1.3 p127] pour le tri. On peut aussi proposer l'algorithme de Dijkstra en version optimisée comme application des files de priorités min 901,903,926,(927), ?

— Borne inférieur du nombre de comparaison d'un tri basé uniquement sur les comparaisons $O(n \log n)$, par la méthode de l'arbre binaire complet des pointeurs successifs sur l'instruction courante (bien plus clair que l'arbre des permutations souvent invoqué), cf [Aho, Hopcroft, Ullman, Ch86 p284]. Exemple du tri à bulle (meilleur cas $O(n \log n)$, pire cas $O(n^2)$) [BBC, ?] ou bien fait dans le [Stern, ?] 903, ?

Autres idées

— Un morceau de l'analyse en moyenne du tri rapide, traité dans [BBC, Ch5.1.1 p122] ou [Aho, Hopcroft, Ullman, Ch8.3 p268] 902,903,926, ?

906. Programmation dynamique : exemples et applications.

— Plus longue sous séquence commune, preuve et un exemple (si le temps, parler du problème similaire du plus long sous mot commun, et ou exhiber la difficulté à la généralisation à n arguments), [Cormen, Ch15.4 341]. On peut citer en application la comparaison de chaînes d'ADN mais aussi la commande `diff` des systèmes GNU/Linux (cf. [Aho, Hopcroft, Ullman, Ch5.6 p192]) 906,907, ?

— Algorithme de Cocke-Kasami-Younger pour résoudre le problème du mot en temps $O(|w|^3)$, par programmation dynamique (attention à l'initialisation). Suppose la grammaire en forme normale de Chomsky, ce qui prend un temps $O(|G|^2)$ et produit une grammaire équivalente de taille $O(|G|^2)$ en partant de G. Bien fait dans [Hopcroft, Ullman, Ch7.4.4, p298] ou esquissé dans [Carton, Ex4.7 Fig4.2 p170] 906,907,910,923, ?

Autres idées

— Algorithme de Floyd pour le calcul du plus court chemin depuis chaque sommet pour un graphe valué dans un semi-anneau étoilable, [Cormen, Ch25.2]. Il faut faire le lien avec la construction de Thomson-Yamada pour le théorème de Kleene (cf. [Carton, p38]). Voir aussi [BBC, Ch4.2.3 p84] ou [Fournier, 1, ?] 906,925,(926), ?

907. Algorithmique du texte : exemples et applications.

— Algorithme glouton de Huffman et codage optimal (preuve d'un des deux lemmes, et un petit exemple), bien traité [Cormen, Ch16.3 p377] ou [Crochemore, Rytter, Ch10.2 p217] 907,926, ?

— Plus longue sous séquence commune, preuve et un exemple (si le temps, parler du problème similaire du plus long sous mot commun, et ou exhiber la difficulté à la généralisation à n arguments), [Cormen, Ch15.4 341]. On peut citer en application la comparaison de chaînes d'ADN mais aussi la commande `diff` des systèmes GNU/Linux (cf. [Aho, Hopcroft, Ullman, Ch5.6 p192]) 906,907, ?

Autres idées

— Algorithme de Cocke-Kasami-Younger pour résoudre le problème du mot en temps $O(|w|^3)$, par programmation dynamique (attention à l'initialisation). Suppose la grammaire en forme normale de Chomsky, ce qui prend un temps $O(|G|^2)$ et produit une grammaire équivalente de taille $O(|G|^2)$ en partant de G. Bien fait dans [Hopcroft, Ullman, Ch7.4.4, p298] ou esquissé dans [Carton, Ex4.7 Fig4.2 p170] 906,907,910,923, ?

— ? Calcul du tableau des bords maximums comme pré-traitement pour l'algorithme de Knuth-Morris-Pratt (correction et terminaison) [Cormen, ?] ou [Crochemore, Rytter, ?] 907,926,927, ?

909. Langages rationnels. Exemples et applications.

— À propos d'intersection de langages : rationnel et rationnel reste rationnel, algébrique et algébrique ne reste pas algébrique (contre exemple $\{a^n.b^n.c^n, n \in \mathbb{N}\}$ par le lemme d'Ogden), et algébrique et rationnel reste algébrique. Esquissé dans [Carton, ?], et bien fait dans [Hopcroft, Ullman, Ch7.3.4 p285] 909,910, ?

— Décidabilité de l'arithmétique de Presburger (seulement le prédicat `=` et le symbole de fonction `+` sur le domaine $\mathbb{N}$) par une construction itérative d'automates déterministes, [Carton, Th3.63 p164] ou [RDavid, ?] 909,914,917,922,924, ?

Théorème de Kleene [Carton, Th1.59 p36] Non c'est trop classique et trop simple.

— ? Tout langage reconnaissable est rationnel (preuve par la construction de McNaughton et Yamada) faire le lien avec l'algorithme de Floyd pour le calcul (matriciel) des plus courts chemins dans un semi-anneau étoilable [Carton, p38] 908,909, ?

— ? Tout langage rationnel est reconnaissable (preuve par la construction de Thomson) 908,909, ?

— ? Tout langage rationnel est reconnaissable (preuve par les automates émondés) [Carton, p36-37] 908,909, ?

910. Langages algébriques. Exemples et applications.

— Algorithme de Cocke-Kasami-Younger pour résoudre le problème du mot en temps $O(|w|^3)$, par programmation dynamique (attention à l'initialisation). Suppose la grammaire en forme normale de Chomsky, ce qui prend un temps $O(|G|^2)$ et produit une grammaire équivalente de taille $O(|G|^2)$ en partant de G. Bien fait dans [Hopcroft, Ullman, Ch7.4.4, p298] ou esquissé dans [Carton, Ex4.7 Fig4.2 p170] 906,907,910,923, ?

— Mises en formes normales **réduite** et **propre**. Application : un algo pour décider si $L_G(S)$ est infini, pour (G, S) donné (revient à décider la présence d'un cycle dans un graphe, par parcours en profondeur puis détection des arcs rétrogrades). Fait rapidement dans le [Carton, ?] ou plus en détail (avec un exemple commun pour chaque étape) dans le [Dehornoy, ?] 910,920,(922), ?

— À propos d'intersection de langages : rationnel et rationnel reste rationnel, algébrique et algébrique ne reste pas algébrique (contre exemple $\{a^n.b^n.c^n, n \in \mathbb{N}\}$ par le lemme d'Ogden), et algébrique et rationnel reste algébrique. Esquissé dans [Carton, ?], et bien fait dans [Hopcroft, Ullman, Ch7.3.4 p285] 909,910, ?

Autres idées

— ? Formes normales, normale quadratique de Chomsky et de Greibach (en faire une bien, les autres en donnant l'idée plus un exemple). Bien fait dans [Carton, ?] mais aussi dans [Dehornoy, App3.20 p110] et [Dehornoy, App3.21 p113], 910, ?

— ? Théorème de Chomsky-Schützenberger (assez dur mais intéressant), [Carton, ?] pour une esquisse, ou [Autebert, Langages et Automates, ?] ou [Autebert, Langages algébriques, ?] 910, ?

912. Fonctions récursives primitives et non primitives (ie récursive).

— Opération arithmétiques récursives primitives (partir des briques de bases, montrer et prouver la somme, prédécesseur, différence ; puis produit et égalité ; puis division et reste ; puis puissance, racine, logarithme entier). Application, `isPrime` est récursive primitive, donc décider si un entier est premier est décidable (même si on sait que c'est polynomial depuis 2002) [Carton, p168-169] ou [Cori1, ?] 912,914, ?

— Toute fonction récursive primitive est M-lente pour un bon M (lemme dans la preuve de Ackermann est récursive mais non primitive récursive). Cf [Cori1, ?]) ou aussi traitée dans [Dehornoy, Par8.3.2 p188]. 912,922, ?

Autres idées

— ? Si une fonction a un temps de calcul récursif primitif alors elle est elle-même récursive primitive [Non Sourcé, ?] 912,913, ?

913. Machines de Turing. Applications.

— Théorème de Savitch, très long dans [Carton, Th4.24 p184] ou dans sa forme générale ($\mathcal{PS} = \mathcal{NPS}$) dans [Hopcroft, Ullman, Th11.5 p477] 913,915,(922), ?

— Théorème de Rice pour les langages reconnus par machines de Turing, bien traité dans le [Carton, ?], ou aussi dans [Hopcroft, Ullman, Th9.33 p387] (912),913,914,(922), ?

Autres idées

— Borne inférieure pour les machines reconnaissant le langage des palindromes [Non Sourcé, ?] 913,915, ?

— Construction d'un langage récursif mais non récursivement énumérable (afin de prouver que l'inclusion est stricte), [Wolper, Ch7 p195] ou aussi [Hopcroft, Ullman, Ch9.2 p372] 913,922, ?

914. Décidabilité et indécidabilité. Exemples.

— Décidabilité de l'arithmétique de Presburger (seulement le prédicat `=` et le symbole de fonction `+` sur le domaine $\mathbb{N}$) par une construction itérative d'automates déterministes, [Carton, Th3.63 p164] ou [RDavid, ?] 909,914,917,922,924, ?

— Théorème de Rice pour les langages reconnus par machines de Turing, bien traité dans le [Carton, ?], ou aussi dans [Hopcroft, Ullman, Th9.33 p387] (912),913,914,(922), ?

Autres idées

— ? Indécidabilité de la logique du premier ordre, [Goubault, ?] ou [RDavid, ?] 914,917,924, ?

— Indécidabilité de la confluence [Baader, Nipkow, Ch6.1 Th6.1.1 p134] par réduction depuis le problème de la provabilité équationnelle. **Attention** dur (même si ce n'est pas très long) ! 914,919,920, ?

915. Classes de complexité : exemples.

— NP-complétude de la recherche d'une clique de taille quelconque dans un graphe, bien fait dans [Carton, p195] par réduction depuis **3SAT**. Faire un exemple avec un dessin d'un petit graphe associé à une petite instance de **3SAT** 915,928, ?

— Théorème de Savitch, très long dans [Carton, Th4.24 p184] ou dans sa forme générale ($\mathcal{PS} = \mathcal{NPS}$) dans [Hopcroft, Ullman, Th11.5 p477] 913,915,(922), ?

Autres idées

— Théorème de Cook-Lévin : **3SAT** est NP-complet (réduction polynomiale depuis SAT) [Carton, p191] (913),915,916,928, ?

— ? Borne inférieure pour les machines reconnaissant le langage des palindromes [Non Sourcé, ?] 913,915, ?

— ? NP-complétude de la recherche de chemin hamiltonien dans un graphe, en opposition à la recherche de chemin eulérien qui est linéaire (cf second projet), bien fait dans [Carton, p193] 915,928, ?

916. Formules du calcul propositionnel : représentation, formes normales, satisfiabilité.

— Mise en forme normale conjonctive (CNF) et disjonctive par un procédé de réécriture dont on montre la correction, la confluence et la terminaison. Éventuellement un exemple de formule dont toutes les CNF sont de tailles exponentielles ($\bigvee_{1 \leqslant i \leqslant n} A_i \wedge B_i$). Application à la lecture de la satisfiabilité ou la validité en temps sous-linéaire. Esquissé dans [Bellot, Sakarovitch, ?] et le système de réécriture est dans [Lalement, ChIII.3 Tab.11] 916,920, ?

— Théorème de compacité en logique propositionnelle, par un argument topologique (théorème général de Thykonov, version non dénombrable), et application au coloriage de graphe, [Goubault, ?] ou [Cori1, ?] 916, ?

Autres idées

— Théorème de Cook-Lévin : **3SAT** est NP-complet (réduction polynomiale depuis SAT) [Carton, p191] (913),915,916,928, ?

917. Logique du premier ordre : syntaxe et sémantique.

— La théorie des ordres denses est décidable, en montrant qu'elle admet l'élimination des quantificateurs [RDavid, Ch3.7.2 p131] ou pas très bien fait dans [Cori2, Ex8.1.7 p192] 917,924, ?

— Décidabilité de l'arithmétique de Presburger (seulement le prédicat `=` et le symbole de fonction `+` sur le domaine $\mathbb{N}$) par une construction itérative d'automates déterministes, [Carton, Th3.63 p164] ou [RDavid, ?] 909,914,917,922,924, ?

Autres idées

— Complétude réfutationnelle de la résolution via le lemme de relèvement, bien fait dans [Stern, Ch7, p243 :249]. (mais aussi peut-être [Goubault, ?] ou [Cori2, ?]) 917,918,919, ?

— ? Indécidabilité de la logique du premier ordre, [Goubault, ?] ou [RDavid, ?] 914,917,924, ?

918. Systèmes formels de preuve en logique du premier ordre : exemples.

— Complétude réfutationnelle de la résolution via le lemme de relèvement, bien fait dans [Stern, Ch7, p243 :249]. (mais aussi peut-être [Goubault, ?] ou [Cori2, ?]) 917,918,919, ?

— Correction, cohérence, compacité et complétude du calcul des séquents du premier ordre (LK_1) [Goubault, ?], à partir des propriétés admises pour le système de Genzhen à l'ordre 0 (LK_0) et le théorème de Herbrand 918, ?

Autres idées

— ? Complétude de la déducation naturelle, [Goubault, ?] ou [Cori2, ?] 917,918, ?

919. Unification : algorithmes et applications.

— Présentation, preuve de terminaison de l'algorithme d'unification naïf et l'intuition de la correction ou bien un exemple, [Goubault, ?] ou [Baader, Nipkow, Ch4.6 p75] 919,(920),927, ?

— Indécidabilité de la confluence [Baader, Nipkow, Ch6.1 Th6.1.1 p134] par réduction depuis le problème de la provabilité équationnelle. **Attention** dur (même si ce n'est pas très long) ! 914,919,920, ?

Autres idées

— ? Complétion de la théorie équationnelle des groupes par le procédé (semi-)automatique de Knuth-Bendix (assez long, prudence). Bien corrigé dans [Lalement, V.3.3 p239] (919),920, ?

— Complétude réfutationnelle de la résolution via le lemme de relèvement, bien fait dans [Stern, Ch7, p243 :249]. (mais aussi peut-être [Goubault, ?] ou [Cori2, ?]) 917,918,919, ?

920. Réécriture et formes normales. Exemples.

— Complétion de la théorie équationnelle des groupes par le procédé (semi-)automatique de Knuth-Bendix (assez long, prudence). Bien corrigé dans [Lalement, V.3.3 p239] (919),920, ?

— Mise en forme normale conjonctive (CNF) et disjonctive par un procédé de réécriture dont on montre la correction, la confluence et la terminaison. Éventuellement un exemple de formule dont toutes les CNF sont de tailles exponentielles ($\bigvee_{1 \leqslant i \leqslant n} A_i \wedge B_i$). Application à la lecture de la satisfiabilité ou la validité en temps sous-linéaire. Esquissé dans [Bellot, Sakarovitch, ?] et le système de réécriture est dans [Lalement, ChIII.3 Tab.11] 916,920, ?

— Indécidabilité de la confluence [Baader, Nipkow, Ch6.1 Th6.1.1 p134] par réduction depuis le problème de la provabilité équationnelle. **Attention** dur (même si ce n'est pas très long) ! 914,919,920, ?

921. Algorithmes de recherche et structures de données associées.

— Structure efficace de dictionnaire, via le hachage parfait, présentation très propre des espaces sur lesquels on fait les calculs probabilistes, comme fait dans le [Cormen, Ch11.5 p238] 901,921, ?

— Arbres binaires de recherche AVL (recherche, insertion et suppression en $O(\log n)$) (structures efficaces de dictionnaire) [Cormen, en exercice non corrigé] et [BBC, Ch6.2 p148] 901,921, ?

922. Ensembles récursifs, récursivement énumérables. Exemples.

C'est une leçon difficile. Il faut réussir à faire quelque chose de différent que la 912 (« Fonctions récursives primitives et non primitives »).

— Construction d'un langage récursif mais non récursivement énumérable (afin de prouver que l'inclusion est stricte), [Wolper, Ch7 p195] ou aussi [Hopcroft, Ullman, Ch9.2 p372] 913,922, ?

— Toute fonction récursive primitive est M-lente pour un bon M (lemme dans la preuve de Ackermann est récursive mais non primitive récursive). Cf [Cori1, ?]) ou aussi traitée dans [Dehornoy, Par8.3.2 p188]. 912,922, ?

— Décidabilité de l'arithmétique de Presburger (seulement le prédicat `=` et le symbole de fonction `+` sur le domaine $\mathbb{N}$) par une construction itérative d'automates déterministes, [Carton, Th3.63 p164] ou [RDavid, ?] 909,914,917,922,924, ?

923. Analyses lexicale et syntaxique : applications.

Leçon difficile. Il faut surtout mettre plein d'exemples, montrer que l'on comprend les utilisations réelles de ces notions. Voir par exemple [Grune et al], [Aho, Ullman] ou encore [Cousineau, Mauny, Ch9].

— Algorithme de Cocke-Kasami-Younger pour résoudre le problème du mot en temps $O(|w|^3)$, par programmation dynamique (attention à l'initialisation). Suppose la grammaire en forme normale de Chomsky, ce qui prend un temps $O(|G|^2)$ et produit une grammaire équivalente de taille $O(|G|^2)$ en partant de G. Bien fait dans [Hopcroft, Ullman, Ch7.4.4, p298] ou esquissé dans [Carton, Ex4.7 Fig4.2 p170] 906,907,910,923, ?

— Analyseurs lexical et syntaxique pour le langage d'une calculatrice minimaliste, exemple d'expressions, d'arbres, de dérivations et d'interprétation (i.e. d'évaluation d'un arbre), [Aho, Ullman, Ch4.9.1 p263] avec yacc et lex 923, ?

— ? Analyse d'une grammaire $LL(1)$, [Aho, Ullman, Ch4.4] attention c'est pas facile ! 923, ?

924. Théories et modèles en logique du premier ordre. Exemples.

— La théorie des ordres denses est décidable, en montrant qu'elle admet l'élimination des quantificateurs [RDavid, Ch3.7.2 p131] ou pas très bien fait dans [Cori2, Ex8.1.7 p192] 917,924, ?

— Indécidabilité de la logique du premier ordre, [Goubault, ?] ou [RDavid, ?] 914,917,924, ?

— Décidabilité de l'arithmétique de Presburger (seulement le prédicat `=` et le symbole de fonction `+` sur le domaine $\mathbb{N}$) par une construction itérative d'automates déterministes, [Carton, Th3.63 p164] ou [RDavid, ?] 909,914,917,922,924, ?

925. Graphes : représentations et algorithmes.

— Algorithme de Dijkstra pour un graphe valué dans le semi-anneau étoilable $\mathbb{R}_{\min}$ à poids positif. Présentation de la version non optimisée, complexité et correction du choix glouton, et explication en conclusion de l'optimisation via file priorité min (tas binaire min). Bien traité dans [Cormen, Ch24.3 p577] avec un exemple en [Cormen, Fig24.6 p578]. Aussi dans [BBC, ?] 921,925,926,(927), ?

— Algorithme de Floyd pour le calcul du plus court chemin depuis chaque sommet pour un graphe valué dans un semi-anneau étoilable, [Cormen, Ch25.2]. Il faut faire le lien avec la construction de Thomson-Yamada pour le théorème de Kleene (cf. [Carton, p38]). Voir aussi [BBC, Ch4.2.3 p84] ou [Fournier, 1, ?] 906,925,(926), ?

926. Analyse des algorithmes : complexité. Exemples.

— Algorithme glouton de Huffman et codage optimal (preuve d'un des deux lemmes, et un petit exemple), bien traité [Cormen, Ch16.3 p377] ou [Crochemore, Rytter, Ch10.2 p217] 907,926, ?

— Tri par tas (en $O(n \log n)$) et explication de la structure de tas binaire (min), complexité de `creerTas` en $O(n \log n)$) [Cormen, ?] et [BBC, Ch3.4 p56] pour les files min et [BBC, Ch5.1.3 p127] pour le tri. On peut aussi proposer l'algorithme de Dijkstra en version optimisée comme application des files de priorités min 901,903,926,(927), ?

Autres idées

— ? Algorithme de Floyd pour le calcul du plus court chemin depuis chaque sommet pour un graphe valué dans un semi-anneau étoilable, [Cormen, Ch25.2]. Il faut faire le lien avec la construction de Thomson-Yamada pour le théorème de Kleene (cf. [Carton, p38]). Voir aussi [BBC, Ch4.2.3 p84] ou [Fournier, 1, ?] 906,925,(926), ?

— ? Algorithme de Dijkstra pour un graphe valué dans le semi-anneau étoilable $\mathbb{R}_{\min}$ à poids positif. Présentation de la version non optimisée, complexité et correction du choix glouton, et explication en conclusion de l'optimisation via file priorité min (tas binaire min). Bien traité dans [Cormen, Ch24.3 p577] avec un exemple en [Cormen, Fig24.6 p578]. Aussi dans [BBC, ?] 921,925,926,(927), ?

927. Exemples de preuve d'algorithme : correction, terminaison.

— Preuve **très méticule** du calcul itératif de la factorielle, en logique de Hoare, cf [Winskel, Exemple Ch6.6 p93] qui semble être le seul livre à traiter un exemple précisément (918),927, ?

— Présentation, preuve de terminaison de l'algorithme d'unification naïf et l'intuition de la correction ou bien un exemple, [Goubault, ?] ou [Baader, Nipkow, Ch4.6 p75] 919,(920),927, ?

— Correction des règles la logique de Hoare, comme fait dans le [Winskel, ?] 927, ?

Autres idées

— ? Algorithme de Dijkstra pour un graphe valué dans le semi-anneau étoilable $\mathbb{R}_{\min}$ à poids positif. Présentation de la version non optimisée, complexité et correction du choix glouton, et explication en conclusion de l'optimisation via file priorité min (tas binaire min). Bien traité dans [Cormen, Ch24.3 p577] avec un exemple en [Cormen, Fig24.6 p578]. Aussi dans [BBC, ?] 921,925,926,(927), ?

— Tri par tas (en $O(n \log n)$) et explication de la structure de tas binaire (min), complexité de `creerTas` en $O(n \log n)$) [Cormen, ?] et [BBC, Ch3.4 p56] pour les files min et [BBC, Ch5.1.3 p127] pour le tri. On peut aussi proposer l'algorithme de Dijkstra en version optimisée comme application des files de priorités min 901,903,926,(927), ?

928. Problèmes NP-complets : exemples de réductions

— Théorème de Cook-Lévin : **3SAT** est NP-complet (réduction polynomiale depuis SAT) [Carton, p191] (913),915,916,928, ?

— NP-complétude de la recherche d'une clique de taille quelconque dans un graphe, bien fait dans [Carton, p195] par réduction depuis **3SAT**. Faire un exemple avec un dessin d'un petit graphe associé à une petite instance de **3SAT** 915,928, ?

Autres idées

— ? NP-complétude de la recherche de chemin hamiltonien dans un graphe, en opposition à la recherche de chemin eulérien qui est linéaire (cf second projet), bien fait dans [Carton, p193] 915,928, ?

Fin leçons info.

Plusieurs références ou autres pointeurs

1. `agreg-cachan.fr/wiki/PlansDeLecons`,
2. `agreg-cachan.fr/wiki/Developpemnts`,
3. `agreg-cachan.fr/wiki/Lecons`,
4. `agreg-cachan.fr/info`,
5. DynaMaths.

Bibliographie presque complète pour l'informatique

On liste ici tous livres cités plus haut. Éventuellement, il faudra essayer de trouver les bouquins qui manquent (ceux qui pointent sur le catalogue de la BU). Les liens sont déjà mis, mais à comparer avec le contenu du dossier ici sur perso.crans.org/besson/agreg/books. Voir la commande `make booksManquant` pour savoir les livres encore à télécharger.

Liste des livres

[TODO] et [FIXME]

Livres génériques

[Dehornoy] « Mathématiques de l'informatique : cours et exercices corrigés »

Un excellent bouquin, qui présente rapidement tous les éléments du programme, avec cours, exemples, démonstrations et exercices ! Contient plein de développements parmi les plus utiles ou les plus classiques.

[Albert, Gastin] « Cours et exercices d'informatique : classes préparatoires »

Un bon livre, niveau prépa. Avec beaucoup de programmes `Caml`, mais peu de preuves. De bons rappels de cours sur les bases, souvent illustrés par un peu de code. Peut vraiment aider pour l'épreuve de modélisation ! (en partie rédigé par Paul Gastin !)

[Stern] « Fondements mathématiques de l'informatique »

Un bon bouquin, mais qui a mal vieilli. Encore de bonnes démos, notamment des réductions pour les problèmes **NP**-complets.

Langages formels et automates

[Carton] « Langages formels, Calculabilité et Complexité »

Une excellente référence. Attention aux quelques fautes (notamment, sur la hauteur d'étoile). Beaucoup de développements, en langages formels bien sûr, mais aussi ailleurs (notamment problèmes **NP** et réductions). Mon livre préféré (parmi ceux pour l'option info) !

[Hopcroft, Ullman] « Introduction to automata theory, languages, and computation »

Une excellente référence (en anglais, mais dans une version internationale assez facile à comprendre). Beaucoup de rappels, d'exemples et de bons dessins à réutiliser pour les plans. De bonnes idées de développements sur les automates (Chap 2), les langages rationnels (Chap 3 et 4), algébriques (Chap 5 et 7), mais aussi de l'indécidabilité (Chap 9) des problèmes **NP** (Chap 10, dont `NodeCover`, `IndependentSets`, et `HamPath`) et une introduction à la classe **Co-NP**.

[Sakarovitch] « Éléments de théorie des automates »

Une bonne référence, même si son style austère rebute un peu. Très complet sur plein de choses hors programmes (youpi), mais aussi plein de choses sur ce qui est au programme des leçons d'info. Plein de développements sur tout ce qui concerne les automates, et un peu plus (PCP Th8.2 p42, des problèmes décidables sur les langages rationnels Prop1.11 p77, etc).

[Salomaa] « Formal languages »

Une autre bonne référence, même si elle commence à vieillir (un peu). J'apprécie ses rappels clairs et illustrés d'exemples pour les notions de grammaires $LR(k)$ et $LL(k)$ (p223). Seul bouquin que je connais qui précise qu'il

existe aussi la notion de grammaires $RR(k)$ et $RL(k)$, même si je n'ai toujours pas compris la différence.

Jean-Michel Autebert : langages formels, et calculabilité

Des bouquins qui commencent à vieillir, mais restent de solides références.

[Autebert, Langages et Automates] « Théorie des langages et des automates »

[Autebert, Transductions] « Transductions rationnelles : application aux langages algébriques »

[Autebert, Langages algébriques] « Langages algébriques »

Décidabilité et calculabilité

Ces questions sont aussi abordées dans des livres cités ailleurs ([Carton], [Sakarovitch] etc).

[Autebert, Calculabilité] « Calculabilité et décidabilité : une introduction »

[Wolper] « Introduction à la calculabilité »

Une excellente référence pour les leçons de calculabilité (des preuves bien rédigés, mais des exercices sans correction) et plein développements possibles : inclusion stricte $\mathcal{R} \subsetneq \mathcal{R}E$, problèmes indécidables sur les grammaires (Ch7.5 p207), etc.

[Sipser] « Introduction to the theory of computation »

Une autre bonne référence pour les leçons de calculabilité.

[Garey, Johnson] « Computers and intractability : a guide to the theory of NP-completeness »

Une référence qui donne une longue liste de problèmes **NP**-complets, qui peut être utile pour toutes les leçons (signaler dans n'importe quel plan que tel ou tel problème est NP-complet est toujours intéressant !). Attention aux preuves qui sont parfois trop succinctes ... ou trop compliquées !

Logique (logic and proof theory)

[Cori1] et [Cori2] « Logique mathématique : cours et exercices corrigés », Tomes 1 et 2

Deux excellents livres, à considérer plutôt comme un seul découpé en deux. De nombreux exercices, des rappels de cours précis et concis, et des démonstrations plutôt claires, mais dont la longueur et la typographie un peu désuète pourront rebuter le néophyte.

[RDavid] « Introduction à la logique : théorie de la démonstration »

Un bon complément aux Cori1 et Cori2 ! Excellente référence pour les développements de logique.

[Goubault] « Proof Theory and Automated Deduction »

Une bonne référence, même s'il n'est vraiment pas facile à prendre en main ! Attention à certaines preuves qui restent fausses, et qui sont irrattrapables (détails dans les preuves des théorèmes de Skolem et Herbrand pour le calcul des séquents du premier ordre LK_1, par exemple).

[Winskel] « The formal semantics of programming languages : an introduction »

Semble être une bonne référence (et la seule) pour la sémantique formelle des programmes, notamment la logique de Hoare et les preuves avec un invariant entre chaque lignes du programmes.

[Lassaigne, de Rougemont] « Logique et complexité »

Peu de contenu utile pour le programme de l'agreg, mais peut éventuellement aider pour aller un peu hors du programme en théorie de la complexité, utile pour la fin du plan sur la leçon complexité. Une édition plus récente (2004) est disponible, en anglais.

[Gochet, Gribomont] « Logique » tomes 1 et 2

Deux bonnes références, mais pas vraiment eu l'occasion de m'en servir.

[Bellot, Sakarovitch] « Logique et automates : option informatique en MPSI et MP »

Un livre qui traite presque entièrement le programme de l'option informatique de prépa, et donc qui peut servir de base pour les plans des leçons en logique et sur les automates. Le seul bouquin que j'ai trouvé qui rappelle les intérêts des formes normales conjonctives (cnf) et disjonctives (dnf), la dnf permettant de lire la satisfiabilité en temps sous linéaire (car parallélisable, en effet $\bigvee_i \bigwedge_j C_{i,j}$ est satisfiable ssi aucun littéral n'apparaît à la fois positivement et négativement dans un des $\bigwedge_j C_{i,j}$). De façon duale la cnf permet de lire la validité.

[Beigel, Floyd] « Le langage des machines : introduction à la calculabilité et aux langages formels »

(Je ne le connais pas du tout.) Cité comme référence pour l'algorithme de Cocke-Kasami-Younger ?

[Pabion] « Logique mathématique »

(Je ne le connais pas du tout.)

Logique en lien avec l'unification et réécriture

Leçon 919, 920 et un peu les leçons de logique.

[Baader, Nipkow] « Term rewriting and all that » par Franz Baader et Tobias Nipkow

THE reference (en anglais) pour tout ce qui touche à la réécriture, donc crucial pour les leçons **919** et **920**.

[Lalement] « Logique, réduction, résolution »

Un bon bouquin qui traite de réécriture et d'unification, mais pas seulement (des rappels sur la déduction naturelle, le théorème de Herbrand etc). On trouvera notamment de bons exemples : fonctions récursives simples calculées par réécriture comme la factorielle ou Ackermann, les règles de dérivation formelles (II.2.1 p66), ou encore le fameux exemple de la théorie équationnelle des groupes (bien fait en V.3.3 p239).

Algorithmique

[BBC] « Éléments d'Algorithmique »

Une excellente référence. **Attention** il devient rare. Contient presque tout, avec plein de dessins et plein de preuves.

[Cormen] « Introduction à l'Algorithmique »

La **bible de l'algorithmicien**, toujours précis et rigoureux pour ses preuves. Il convient de rester vigilant, quelques typos ou erreurs restent présentes, même dans la dernière édition. Certaines peuvent inspirer des développements, et certains algorithmes (hachages, arithmétique, ?) peuvent être présentés directement en développement de maths.

[Aho, Hopcroft, Ullman] « Structures de données et algorithmes »

Une excellente référence (plus souvent disponible en anglais). De très bons rappels sur « tout », en particulier les questions de dictionnaires, graphes (orientés ou non), et les tris. Une preuve presque claire de la borne inférieure du nombre de comparaisons pour un algorithme de tri par comparaisons.

[Froidevaux] « Types de données et algorithmes » volumes I et II

Une référence intéressante, en deux volumes. Rédaction et typographie purement illisible, dommage.

[Baynat] « Exercices et problèmes d'algorithmique » (146 énoncés avec solutions détaillées)

Un bon bouquin, qui utilise souvent une rédaction « type développement » et un découpage par lemmes. Notamment les chapitres 5 sur les bases des graphes, 6 sur les parcours et 7 sur les graphes valués.

[Chabert] « Histoire d'algorithmes : du caillou à la puce »

Surtout intéressant pour l'aspect historique de certains domaines de l'algorithmique. Notamment utile pour la méthode de Héron, la méthode de Gauss, etc.

[Boissonnat, Yvinec] « Géométrie algébrique »

Presque trop complet... Utile pour les questions de triangulations et les diagrammes de Voronoï. Présente plein de méthodes de calculs de l'enveloppe convexe (autre que Graham et Jarvis).

Algorithmique du texte

[Crochemore, Rytter] « Text algorithms »

En anglais. Semble distribué en ligne légalement. Présente KM, KMP, les automates de Simon, mais aussi Boyer-Moore (qui est l'algorithme effectivement utilisé dans GNU grep), ainsi que le codage de Huffman, parmi d'autres choses.

Graphes

[Gondran, Minoux] « Graphes et algorithmes »

Un peu vieux, mais reste très complet. Présente les questions de connexité (et calcul des composantes connexes), de problème du plus court chemin (Moore-Dijkstra, Dijkstra, Bellman, Ford, Floyd, Dantzig etc), un bon chapitre sur les matroïdes, et un autre sur les arbres et arborescences (Kruskal, Prim).

[Fournier, 1] et [Fournier, 2] « Graphes et applications : volumes 1 et 2 »

Deux très bons bouquins, orientés applications. Le tome 1 en particulier est très clair et complet avec de bons rappels sur les définitions, les questions de représentations des graphes, mais aussi la recherche de chemins optimaux, de d'arbres couvrants. Le tome 2 présente notamment le problème de voyageur de commerce.

Compilateurs, analyses lexicale et syntaxique

[Aho, Ullman] « Compilateurs : principes, techniques et outils »

Le livre le plus célèbre pour ces questions de compilation, d'analyses lexicale et syntaxique. Quelques exemples très bien traités, dont celui du **Si** .. **Alors** .. **Sinon** en Ch4.3.2 p193, ou celui d'une calculatrice simpliste avec yacc et lex en Ch4.9.1 p263 (qui peut faire un développement).

Ce livre est initialement en anglais, et est surnommé « le dragon » (pour le dessin en couverture).

[Grune et al] « Compilateurs : cours et exercices corrigés »

Un excellent ouvrage, dont la portée dépasse largement le cadre du programme de l'option info. On appréciera notamment son très précieux « résumé » sur les concepts de l'analyse lexicale (2.4, p176) et syntaxique. Il rappelle aussi l'idée qui se cache derrière les termes d'analyses descendante ($LL(1)$) et ascendante ($LR(0), LR(1), SLR(1), LARL(1)$) au Ch2.2 p110. Donne des exemples, parfois agrémenté d'extraits d'implémentations concrètes (avec yacc ou bison).

[Cousineau, Mauny] « Approche fonctionnelle de la programmation »

Semble aborder rapidement les systèmes de types « à la ML » (tirés du λ-calcul), ainsi que les questions d'analyses lexicale et syntaxique.

[Lex et Yacc] « Lex et Yacc »

Un livre traitant surtout des outils `GNU` lex et yacc (pour coder en `C`). Il faudrait la même chose pour OCamlLex et OCamlYacc ([Appel] tente de combler se vide, sans réel succès) !

[Appel] « Modern compiler implementation in ML »

(Je ne le connais pas du tout.)

[Tennent] « Principles of programming languages »

(Je ne le connais pas du tout.)

Cryptographie

[Meunier] « Algèbre avec applications à l'algorithmique et à la cryptographie »

Un très bon bouquin, rappelle les bases sur le cours en algèbre mais va assez loin sur les applications (Diffie-Hellman, RSA, El-Gamal, codes correcteurs, Berlekamp, pseudo-inverse, FFT, et même Miller-Rabin).

[Menezes] « Handbook of applied cryptography »

Une excellente référence (en anglais) pour tout ce qui concerne la cryptographie. Un peu obscur et pas très clair sur les preuves, mais de bons schémas, des exercices et plein d'exemples (de tout, notamment Diffie-Hellman, RSA, ou El-Gamal).

Algèbre et utilisation en informatique (?)

[Demazure] « Cours d'Algèbre »

Un bon bouquin, clair et précis. Beaucoup de contenu sur les codes correcteurs, et très orienté algorithmes et informatique (plus de 100 programmes ruby sont inclus dans le livre !).

Remarque sur la forme et le style de rédaction demandé

Cette liste est bien faite. Pour la modifier ou la compléter, il faut indiquer pour chaque développement :

1. la **liste des leçons concernées**, sous forme de `num1,num2,..,numN,?`, par ordre croissant. Éventuellement rajouter les leçons pour lesquelles on n'est pas sûr ou pas bien convaincu entre parenthèse : `num1,(num2),num3,?`. Rajouter un `,?` final si pas encore sûr de ne pas pouvoir le mettre ailleurs ;
2. **le(s) livre(s)** avec numéro d'exercice et page(s) de **référence**, sans s'embêter avec l'édition, sous forme de lien `[[Auteur,Auteur,Livre], Th4.9, p171]`, en ajoutant la référence à la fin. Mettre un `[[Auteur,Auteur],?]` si plutôt sûr de la référence mais pas vérifié ou pas noté l'endroit précis. Mettre un `[[Non Sourcé],?]` si pas de référence, ou un `[[Auteur?],?]` si le livre n'a pas été rajouté dans la bibliographie ;
3. éventuellement **un lien** vers un (ou plusieurs) **fichiers PDF**, sous forme de lien `[Nom du dév](http://perso.crans.org`
   Rajouter les versions alternatives, au moins celles qui sont bien faites, sous la forme `(ou [2](http://perso.crans.org/be`

