Lang Comp

Un compilateur est un programme qui prend ĉ donnée un programme écrit ds un langage source et le traduit ds un langage appelé langage objet

---

Analyseur lexical → analyse les mots
→ a la structure d'un automate fini qui reconnait des identificateurs.

Analyseur syntaxiq → check la phrase.
→ crée un arbre selon la règle de gramm.
→ vérifie les règles de gramm.

C1

Une grammaire algébriq est donnée par:
- un alphabet terminal A
- ______ de variables
- un ensbl de règles de product° R de la forme $\Theta \to \alpha$

. Si 2 gramm génèrent le m̂ langage, elles st équivalentes.

. Automates finis st des gramm algébriq particulières

. Ambiguïté ⇒ une gramm est ambiguë ssi ∃ un mot ∈ au langage tq on a au ⊖ 2 arbres de dérivat°.

C2

$$\boxed{\begin{array}{l} S \to S\alpha \mid \beta \\ \Rightarrow S \to \beta S' \\ \quad S' \to \alpha S' \mid \varepsilon \end{array}}$$

1. Eliminer la récur gauche
2. Calculer les ensbl FIRST et FOLLOW.
3. Remplir la table d'analyse
4. Executer une analyse LL

STACK | INPUT | OUTPUT

LR(0) → si la table ne contient pas de conflit.
⇒ analyse ascendantes déterministes

LL(1) → s'il y a au ⊕ 1 product° ds chaque case.
⇒ analyse descendantes déterministes

G est LL(1) si ∀ paires de règles distinctes de G de type : $A \to \alpha \mid \beta$:
1. Il n'y a pas de terminal "a" tq α et β dérivent ts deux des chaînes commençant par "a".
2. α et β ne dérivent pas la chaîne vide.
3. Si β dérive ε, alors α ne dérive pas de chaîne commençant par Foll(A)

SLR(1) → si pas de conflits shift/red.
LL(1) → si 1 seul elt ds chaq case, et pas ambiguë et sans récu. gauche.

Passeur SLR
1. Ajouter S'
2. First / Foll.
3. Faire l'automate.
4. Table d'analyse
5. Execut°

STACK	INPUT	OUTPUT
0	—$	

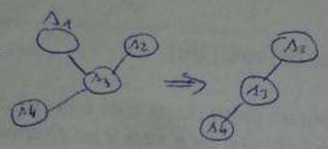
TG | 1

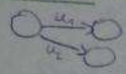
• graphe orienté → ensbl fini de paires ordonnées de sommets, arcs.

• graphe non-—— → ——————— de sommets, arêtes.

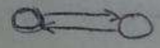
• boucle → 

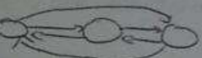
• graphe simple → si sans boucle et si au ⊕ une arête entre 2 sommets.

• Sous-graphe → 

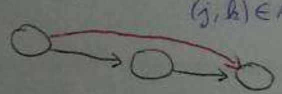
• Deux arcs st dits adjacents s'ils ont au ⊖ une extrémité commune. 

• Deux sommets ———————————— ∃ une arête les joignant

• Graphe sym = 

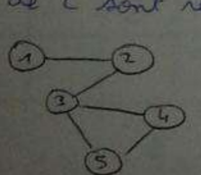
• Graphe complet → si ∀ couple de sommets (x,y), ∃ un arc $x \to y$. 

• Transitif → Si $\left.\begin{matrix}(i,j)\in A \\ (j,k)\in A\end{matrix}\right\}$ Alors $(i,k)\in A$.

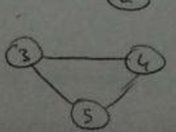


• $\underbrace{d^\circ(x) = \underset{\text{sortant}}{d^{\circ +}(x)} + \underset{\text{entrant}}{d^{\circ -}(x)}}$ = nb d'arcs.

• Clique → tt ensbl de sommets C tq 2 sommets qcq de C sont reliés par une arête



Clique 1 : 

Clique 2 : 

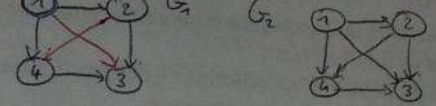
• Mat d'incidence → lg sommet, col arc.
→ {-1 ; 0 ; 1} (pr non-orienté {-1 ; 0})

• Mat d'adjacence → booléens.

• Listes

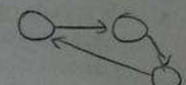
• Applicat° multivo° : $\Gamma(s_1) = s_5$ $\Gamma^-(s_1) = s_2$
ensbl des |succ / préd| de i si |$\Gamma(i)$ / $\Gamma^-(i)$|

• Γ équivalent : deux graphes st ≡ s'ils ont la m̂ fermeture transitive

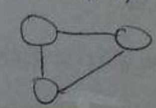


Connexité

• Un graphe orienté est dit forten^t connexe si ∀ paire ordonnée de sommets distincts (u,v), ∃ un chemin de u vers v et un de v vers u.



• Un —— non-orienté —— connexe, si ∀ paire de sommets distincts (u,v), ∃ une chaîne reliant u,v.



• Graphe ss circuit

méth → Tant que supprimer un sommet sans prédécesseur possible. Si on supp tout ✓.

• coût cumulé d'un chemin → la Σ des coûts des arcs qui compose le chemin.

• les circuits de coût < 0 st appelés circuits absorbants

* Dijkstra

→ Calcule le + court chemin entre source S et ts les sommets accessibles depuis S.

→ ⊗ si circuits absorbants.

* Bellman

→ Donne le + court chemin d'un sommet source à ts les autres sommets.

→ Valable ∀ circuit

* Fct rang

→ classe les sommets suivant un ordre bien précis.

* Floyd.

→ recherche d'un + court chemin entre ts les couples de sommets.

TAG
28-11-11

Chap. 2 Plus courts chemins

On présente dans ce chapitre un problème typique de cheminement dans les graphes : **la recherche d'un plus court chemin entre deux sommets.**

Ce problème a des nombreuses applications :

- Trouver le moyen le plus économique pour aller de Brest à Lyon, connaissant pour chaque ligne aérienne le prix de billet d'avion.
- Les problèmes d'optimisation de réseaux (réseaux routiers ou réseaux de télécommunications
- Les problèmes d'ordonnancement
- Les problèmes d'intelligence artificielle tels que la circulation dans un labyrinthe

3.1 Définition, conditions d'existence et exemples

- G = [S, A] est **valué** ⇔ à chaque arc **u** est associé une longueur (un coût) **l(u)**.

- On appelle coût cumulé d'un chemin μ (ou tout simplement coût ou poids de μ) noté coût(μ) ou l(μ), la somme des coûts des arcs qui le compose. On appelle plus court chemin de x vers y un chemin allant de x à y de coût minimum. Un tel chemin n'existe pas toujours.

$$l(\mu) = \sum_{u \in \mu(x,y)} l(u)$$

- La longueur l(u) peut s'interpréter différemment dans de nombreuses applications pratiques.
  - Carte routière et chemin de moindre coût
  - Construction d'une autoroute entre 2 villes

Conditions d'existence :

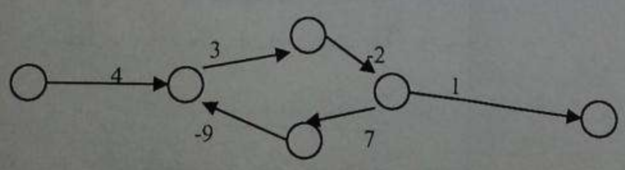
Considérons un chemin μ allant de x à y; supposons ce chemin contient un circuit Γ. Soit μ' le chemin obtenu en supprimant ce circuit de μ. On a :

$$\text{coût}(\mu) = \text{coût}(\mu') + \text{coût}(\Gamma)$$

Si coût(Γ) est négatif, il n'existe pas de plus court chemin de x à y, car étant donné un chemin λ de x à y, on peut toujours construire un chemin λ' de coût strictement inférieur à λ, en passant une fois de plus dans Γ.

Si coût(Γ) est positif ou nul, μ' est de coût inférieur ou égal à μ; c'est un meilleur candidat que μ pour être solution du problème si coût(Γ) est strictement positif.

Les circuits de coût négatif sont appelés **circuits absorbants.**



Dans la suite on suppose qu'il n'existe pas de tels circuits. Un plus court chemin ne peut contenir un circuit de coût positif. S'il y a des circuits de coût nul, il y a plusieurs solutions au problème de la recherche d'un plus court chemin.

Théorie des G

Dijkstra :

CC	M	0 1 2 ...
0	12 ...	0;0

On prend le π le + court des M, qu'on stock dans CC. Et recalcule.
∅ pour arcs<0.

Bellman :

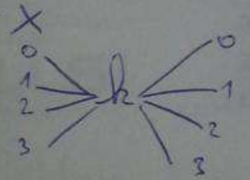
k	0	1 X
0	0	∞ ∞ – ∞
1	0	valeur;0 .....
⋮		
X		

On stop au nb de sommets ou qd plus de chgt. On recalcule les chemins les + court à cha^a k.
Pbl si circuit absorbant

Floyd :

L_k

	0	1	2	3
0	0			
1		0		
2			0	
3				0



P_k

	0	1	2	3
0	0			
1		1		
2			2	
3				3

→ X=0;Y=3

$P_k[X,Y]=P_{k-1}[k,Y]$

L→ donne les chemins les + courts.
P→ —— les prédecesseurs.

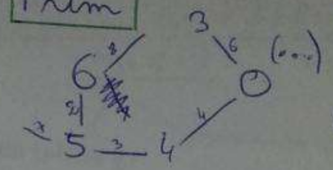
⇒ On s'arrête quand on a étudié tous les sommets.

Kruskall

Destructif → on supp le max tant que le graphe reste connexe.

Constructif → on crée en commençant par les petites valeurs. On stope à n−1 arêtes, avec n nb de sommets. Tant que pas de circuit.

Prim



On choisit un sommet, puis on prend les + petites arêtes.

Ordonnancement.

	Date au + tôt	Date au + tard
Deb	0	
A	0	
B	A+1	
C	max(5, B+2) = 5	min( ) − durée(c)
⋮		$\boxed{D^{+tard}(x) > D^{+tôt}(x)}$
ω		

Intervalle de flottement

↳ $m_i = D^{+tard}(i) - D^{+tôt}(i)$

signal → mesure d'une variat° Ψ porteuse d'une informat°.

DSP → Densité Spectrale de Puiss ⇒ répartit° en fct de la fréq de la puiss du signal

Un signal numéri° dont la fréq centrale est
↳ zéro est dit en bande de base, codage en ligne
↳ f_0 ———— transposée, modulat°

filtre = invariant ds le tps, linéaire et continu.

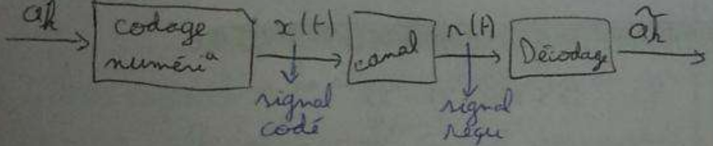
La convolut° devient une multiplicat° ds le domaine fréquentiel.

Intérêts transmission numéri° = permet la mise en place de techni° de :
- cryptographie (sécurité)
- correct° d'erreur (codage canal).

C1 | I.

Canal idéal → ∅ filtrage, juste du bruit

Transmission en bande de base



Le codage de voie, ou numéri°, ou en ligne est la fct qui permet de créer un signal $x(t)$ à partir d'une suite de symb a_k

$$\boxed{x(t) = \sum_k a_k\, g(t-kT)}$$

Débit brut → nb de symb par unité de tps (en bauds)
Débit bin° → si les a_k prennent 2 valeurs. $= \frac{\log_2(M)}{T}$

Bruit blanc → couvre tt les composantes fréquentiel

· Opérat° de codage équivaut à un filtrage

$$\boxed{S_{xx}(f) = E(|X(f)|^2)}$$

a_k indép ⇒ Benett simplifiée

$$\boxed{S_{xx}(f) = |G(f)|^2 \left[\frac{\sigma_a^2}{T} + \frac{m_a^2}{T^2} \sum_n \delta\left(f-\frac{n}{T}\right)\right]}$$

NB : Pr que le spectre de $x(t)$ soit auto-porteur de l'horloge, il faut :
- $m_a \neq 0$ - $G(1/T) \neq 0$.

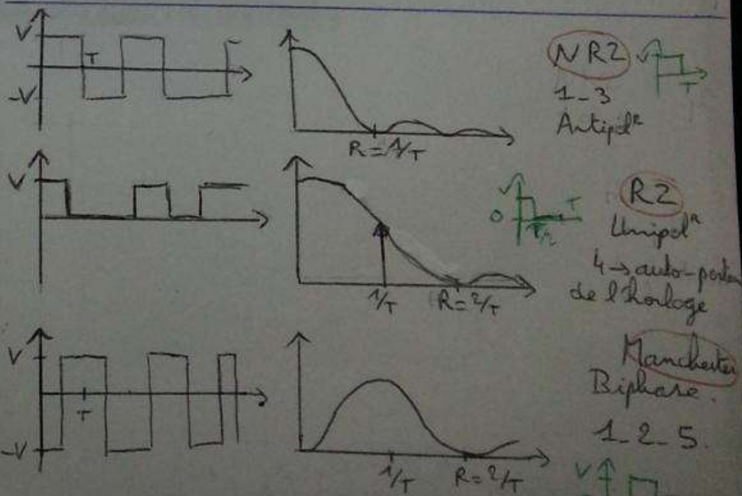
Le spectre du signal :

1. ∅ doit comporter de composante continue.
2. Doit tendre vers 0 qd $f \to 0$.
3. —— avoir un support le + étroit possible
4. —— des raies à la fréq de l'horloge
5. —— conserver l'horloge à court terme

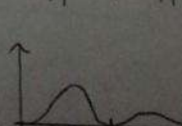
R → rapidité de modulat°, l'inverse du tps le + court pdt lequel le signal reste cste.

valence → nb de valeurs que prennent les symb a_k.

- unipol^R → valeurs ≥ 0 ou nulles
- Antipol^R → —— symétri° ss la valeur 0.
- Bipol^R → ———— et ————.

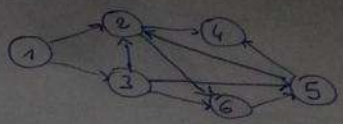


AMI : symb ∅ indép
1. 2. 3



2B1Q { ✓ : débit bin^R + important (occupat° spectrale ...)
⊗ : + sensible au b...

Résultat du parcours d'exemple.



k=0
$S_0 = \{1\}$
CC = {1}
$S_1 = \{2, 3\}$
$\pi(2) = 1$
$\pi(3) = 1$
CC = {1, 2, 3}

k=1
$S_1 = \{2, 3\}$
CC = {1, 2, 3}
$S_2 = \{4, 5, 6\}$
$\pi(4) = 2$
$\pi(5) = 2$
$\pi(6) = 2$
CC = {1, 2, 3, 4, 5, 6}

3.5 Graphe G et longueurs l(u) quelconques : algorithme de Bellman

Cet algorithme est valable pour des graphes sans circuits négatifs, valués par des coûts quelconques.

Il donne un plus court chemin d'un sommet source à tous les autres sommets.

- Soit $\pi(i)$ la longueur du plus court chemin de 1 à i, et l'arc (i, j) est de longueur l_{ij}.
- Pour le plus court chemin de 1 à j pouvant passer par i, on doit avoir :

α $$\pi(j) \leq \pi(i) + l_{ij} \qquad \Rightarrow \qquad \pi(j) - \pi(i) \leq l_{ij}$$

- π peut donc être considéré comme un potentiel : la différence de potentiel entre i et j doit être inférieure ou égale à l_{ij}.

Principe

- Soit G=[S, A] un graphe dont les arcs sont munis de longueurs réelles quelconques.
- Lorsque la procédure se termine, $\pi(i)$ représente la valeur du plus court chemin entre 1 et i.
- Les $\pi(i)$, $\forall i \in S$ sont modifiés itérativement de façon à satisfaire à la condition d'optimilité suivante :

  Un ensemble de valeurs $\pi^*(i)$, i = 1, 2, …, N et $\pi^*(1) = 0$ représente les longueurs des plus courts chemins de 1 aux autres sommets de G $\Leftrightarrow$ ces valeurs vérifient l'ensemble des inéquations suivantes :

  $$\forall (i, j) \in A \ \pi^*(j) \leq \pi^*(i) + l_{ij}. \ (****)$$

- L'idée : parcourir séquentiellement la liste de tous les arcs et vérifier la condition (****)

II_ Canal Idéal

(ds notre cas).

$n(t)$ = Bruit Blanc Gaussien Additif.

* -> réparti aléatoiren^t suivant une var. gaussien

- Le filtre de récept° $g_r(t)$ permettra de limiter l'impact du bruit sur la proba d'erreur
- L'échantillonnage de $r(t)$, ts les kT, permettra d'estimer les $\hat{a}_k$.
- Erreur bin^e qd $\tilde{a}_0 \neq a_0$.

Filtre adapté

$$\boxed{P_e = \frac{1}{2}\,\text{erfc}\sqrt{\frac{E_b}{N_0}}}$$

E_b : nrj moyenne par bit.

$N_0/2$: DSP bilatéral du bruit

Filtre adapté $\rightarrow P_e = \frac{1}{2}\,\text{erfc}\sqrt{\frac{E_{mb}}{2N_0}}$

III

- filtrage peut entraîner IES.

$$a(t) \rightarrow \boxed{G(f)} \xrightarrow{x(t)} \boxed{\text{Canal } C(f)} \xrightarrow{r(t)}$$

$H(f) = C(f) \times G(f)$.

L'IES

Canal idéal -> le signal ne dépend que de a_p.

—— filtrant -> —— dépend des précédents

* Pr éviter l'IES
  - il faut ↓ le débit bint.
  - on va imposer à $h(t) = 0$ ts les T.

Critère de N:

$$h(t) \text{ doit ê tq } \exists\, t_0 \text{ tq } \begin{cases} h(t_0) \neq 0 \\ h(t_0 + nT) = 0 \quad \text{pr } n \neq 0 \end{cases}$$

∀ les fcts de transfert qui rep au critère de N, la bande passante offerte par le canal ? doit ê au ⊖ égale à $1/2T$.

D. de l'œil :

verticalem^t -> marge d'erreur liée au bruit add.

horizontalem^t -> ——— sur l'échantillonnage.