Programmation linéaire
TD modélisation IP

Laurent Lemarchand
Lab STICC/UBO
Laurent.Lemarchand@univ-brest.fr

TD programmation lineaire
Allocation de ressources de calcul (ex. Amazon WS)

- Vente de temps de calcul sur des machines physiques
  - Machines physiques PM_j caractérisées par
    - Leur cout énergetique e_j ($ par unité de calcul)
    - Leur capacité de calcul c_j (en unités de calcul)
    - Leur cout fixe de démarrage f_j ($)
  - Les calculs correspondent aux lancements de VM
    - Nombre d'unités de calcul u_i nécessaire pour chaque VM_i
- Comment placer les VM_i sur les PM_j ?
  - Eventuellement, ressources limitées en PM de chaque type
  - Fonctions objectives possibles

TD programmation lineaire
Allocation de VM (Amazon WS)

- Variables de décision $X_{ij} \rightarrow VM_i$ placée sur PM_j

- Q1 : les VM sont placées sur UNE PM

  Q2 : la capacité d'acceuil des PM doit être respectée

TD programmation lineaire
Allocation de VM (Amazon WS)

- Variables de décision $X_{ij} \rightarrow VM_i$ placée sur PM_j
  - $\forall i, \sum_j X_{ij} = 1$
  - $\forall j, \sum_i u_i.X_{ij} \leq c_j$

- Fct obj : **cout énergétique**

Q3 : définir Y_j, à 1 si un VM i est placée sur la PM j

Q4 : en déduire la fonction de coût énergétique

TD programmation lineaire
Allocation de VM (Amazon WS)

- Variables de décision $X_{ij} \rightarrow VM_i$ placée sur PM_j
  - $\forall i, \sum_j X_{ij} = 1$
  - $\forall j, \sum_i u_i.X_{ij} \leq c_j$

- Fct obj : **cout énergétique**
  - $Y_j = 1$ si machine j démarrée, i.e. Si $\exists X_{ij} > 0$

    $\rightarrow \forall (i,j), X_{ij} \leq Y_j$

    $\rightarrow$ ou modif contrainte de base : $\sum_i u_i.X_{ij} \leq c_j.Y_j$
  - *min* $\sum_j f_j.Y_j + \sum_i \sum_j u_i.e_j.X_{ij}$

TD programmation lineaire
Allocation de VM (Amazon WS)

- Variables de décision $X_{ij} \rightarrow VM_i$ placée sur PM_j
  - $\forall i, \sum_j X_{ij} = 1$
  - $\forall j, \sum_i u_i.X_{ij} \leq c_j$

- Fct obj : si les besoins peuvent évoluer → solution robuste ?
  → **Maximiser la laxité** (qui est le *min* L_j)

  → *max* Lax

Q5 : comment définir la laxité L_j de la VM j ?

Q6 : comment définir Lax (= $\min_j(L_j)$) ?

TD programmation lineaire
Allocation de VM (Amazon WS)

- Variables de décision $X_{ij} \rightarrow VM_i$ placée sur PM_j
  - $\forall i, \sum_j X_{ij} = 1$
  - $\forall j, \sum_i u_i.X_{ij} \leq c_j$

- Fct obj : si les besoins peuvent évoluer → solution robuste ?
  → **Maximiser la laxité** (qui est le *min* L_j)

  → *max* Lax
  - $\forall j, L_j = c_j - \sum_i u_i.X_{ij}$
  - $\forall j, 0 \leq Lax \leq L_j$

TD programmation lineaire
2ème ligne de tram

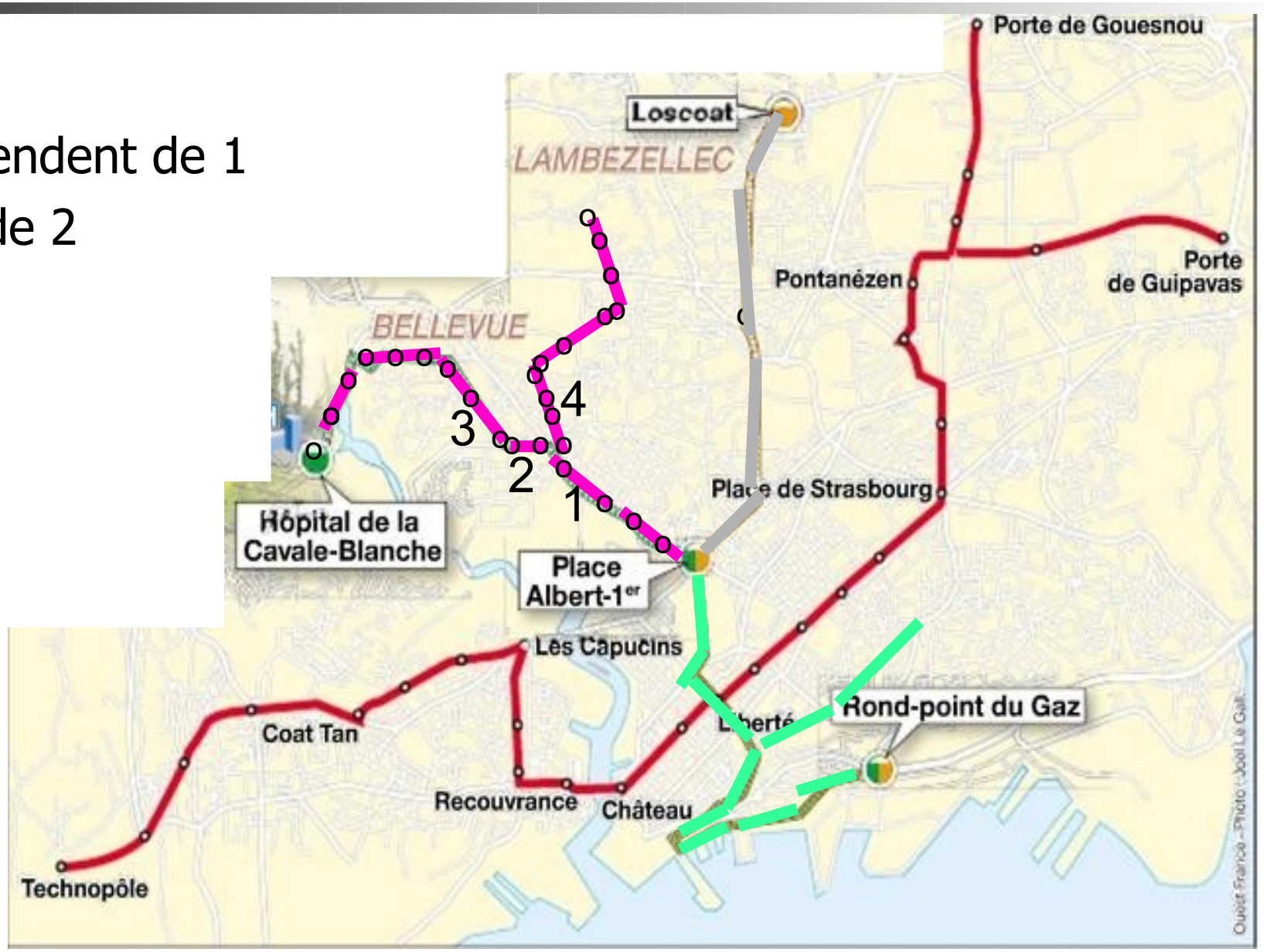
On envisage la construction d'une deuxième ligne de tram

Différents tronçons et parcours sont envisagés, avec pour chacun, un ensemble de stations possible. Chaque station dessert une partie de la population. Les stations ne peuvent etre trop proches les unes des autres. On veut déservir au moins nb habitants

- Tronçons de lignes T_j caractérisés par
  - Leur cout c_j
  - Les stations qui peuvent s'y implanter stations(j)
  - Les tronçons dont ils dépendent deps(j)
- Stations S_i caractérisées par
  - Les points de population qu'elles desservent pop(i)
  - Les stations qui ne peuvent etre construites en meme temps exclues(i), car elles seraient trop proches

TD programmation lineaire
2ème ligne de tram

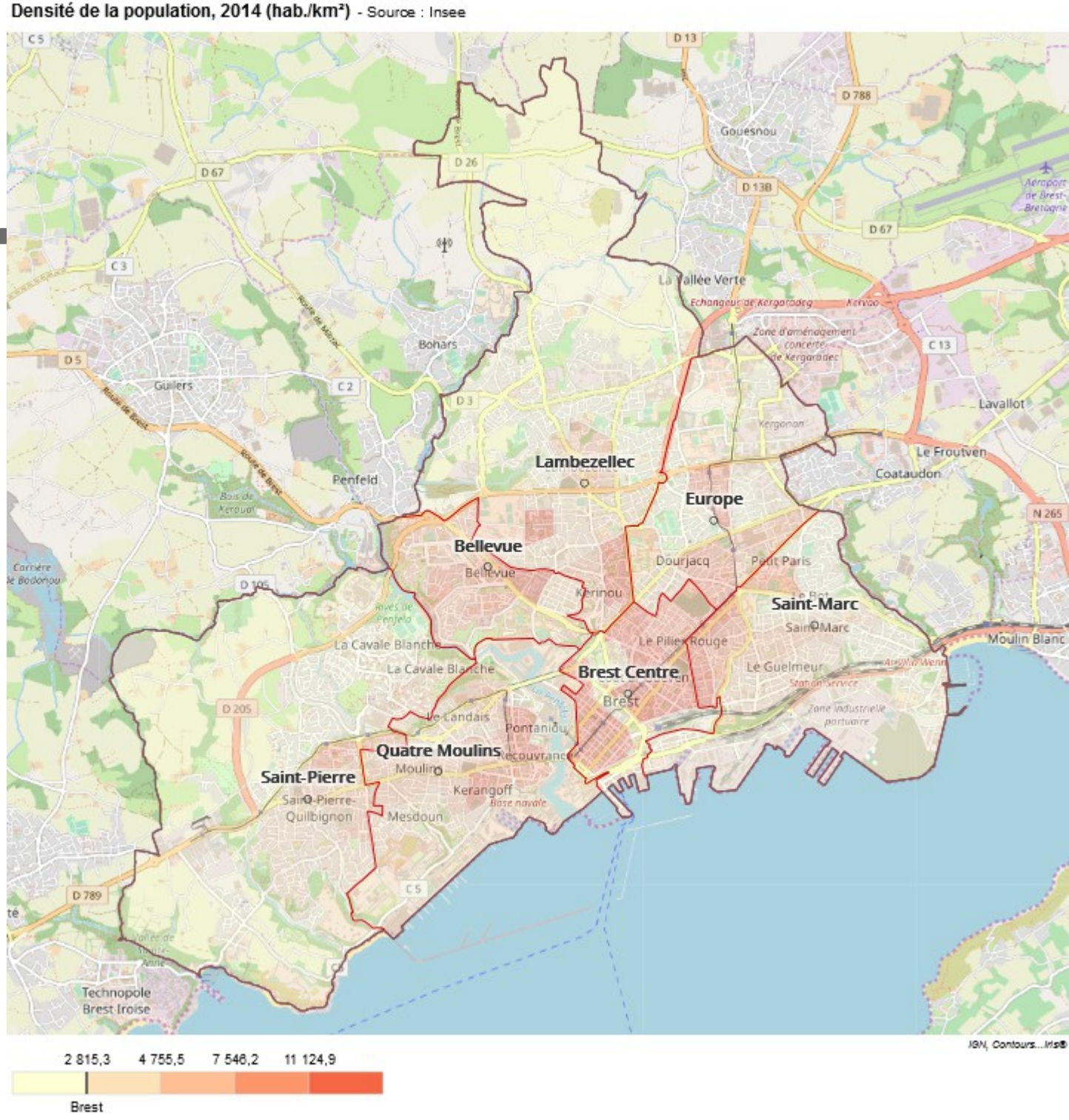
- Tronçons
  - 2 et 4 dependent de 1
  - 3 dépend de 2



TD programmation lineaire

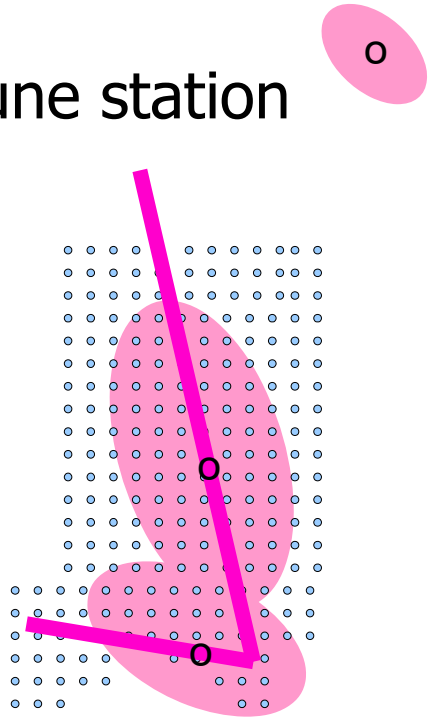
2ème ligne de tram

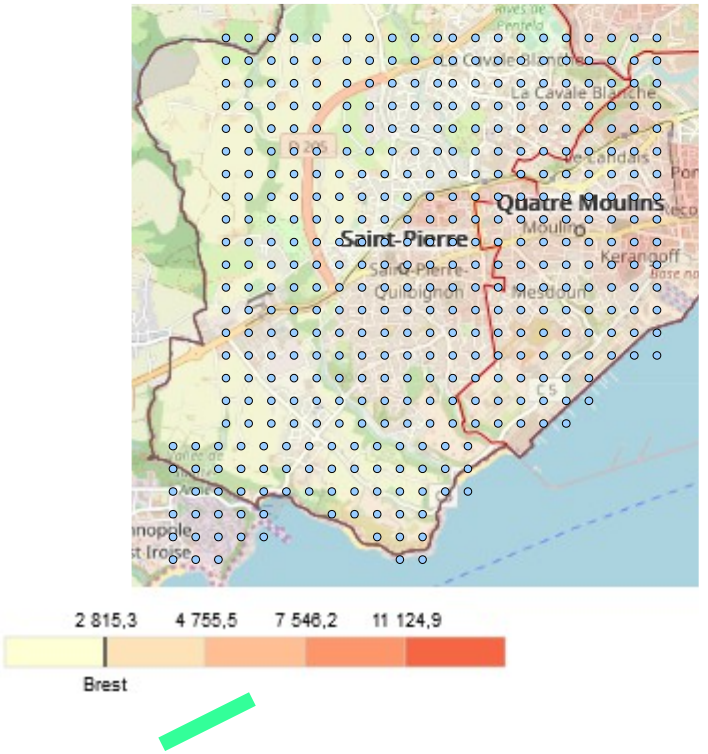
- Densité de population



TD programmation lineaire
2ème ligne de tram

- Densité de population
  - Points P_h avec nombre d'habitants associés n_h
  - Desservis par une station





TD programmation lineaire
2ème ligne de tram

- Comment modéliser le fait que les tronçons dépendent les uns des autres

  implication

- Une station ne peut être construite que si le troncon associé est construit

  implication

- Une station ne peut être construite en même temps que d'autres

  Exclusion mutelle

- P_h est il couvert ?

  implication

TD programmation lineaire
2ème ligne de tram

- Comment modéliser le fait que les tronçons dépendent les uns des autres

  $\forall j, \forall k \in deps(j), T_k \leq T_j$

- Une station ne peut être construite que si le troncon associé est construit

  $\forall j, \forall i \in stations(j), S_i \leq T_j$

- Une station ne peut être construite en même temps que d'autres

  $\forall i, \forall k \in exclues(i), S_i + S_k \leq 1$

- P_h est il couvert ?
  - $\forall i, \forall h \in pop(i), P_h \leq S_i$

TD programmation lineaire
2ème ligne de tram

- Le nombre d'habitants desservis doit être suffisant (seuil nb)

  Population couverte >= nb

- Fonction objective → coût des travaux

  Min stations et tronçons

TD programmation lineaire
2ème ligne de tram

- Le nombre d'habitants desservis doit être suffisant (seuil nb)

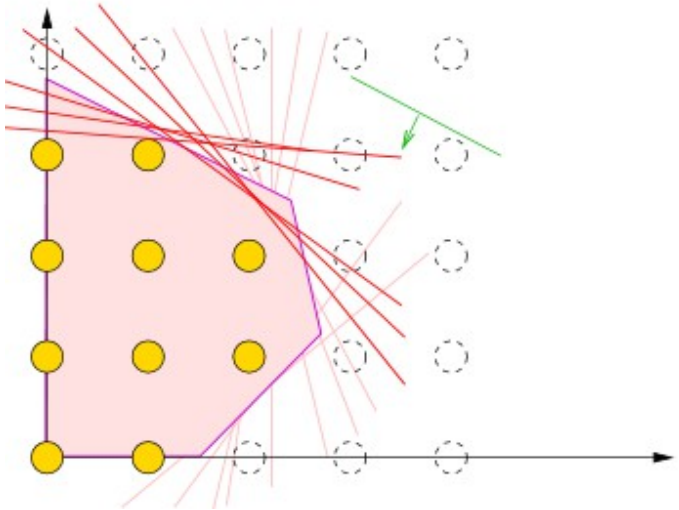
  $\sum_h n_h.P_h \geq nb$

- Fonction objective → coût des travaux

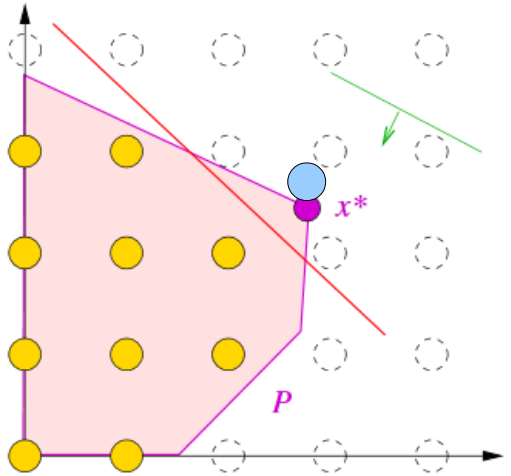
  min $\sum_j c_j.T_j + s.\sum_i S_i$

TD programmation linéaire
Branch and Cut

- Pour un problème décrit par un ensemble trop complexe de contraintes
  - Commencer par une version simplifiée
  - Ajouter de nouvelles contraintes au fur et à mesure (boucle)
  - Illustration : sol valide ; sol du pb simplifié, mais invalide → coupe



problème complet



résolution simplifiée

TD programmation linéaire
Branch and Cut

Voyageur de commerce symétrique, N villes, $D[i,j] = D[j,i] = d_{ij}$

- Passer une et une seule fois par chaque ville en rebouclant
- Var de décision X_{ij}, $i > j$ $\rightarrow$ on va de la ville i à la ville j

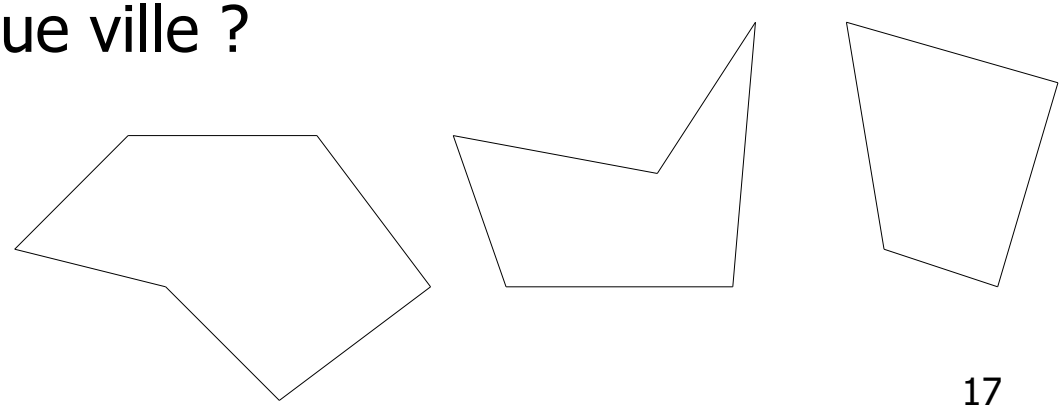
- Fonction objective ?

  $min \sum_{i,j} d_{ij}.X_{ij}$

- Contrainte de passage par chaque ville ?

  $\forall i, \sum_{j} X_{ij} = 2$

  $\rightarrow$ sous-tournées



TD programmation linéaire
Branch and Cut

- Nombre de sous tournées exponentiel dans graphe G = (V, E)

  $\forall W \subset V\ (W \neq V),\ \delta(W) \geq 2 \quad (1)$

  $\delta(W) = \sum_{i,j} X_{ij}$ avec $i \in W, j \notin W$

  - *Il faut une arete qui sort du sous-groupe W*
- Ou

  $\forall W \subset V\ (W \neq V),\ ,\ e(W) < |W| \quad (2)$

  $e(W) = \sum_{i,j} X_{ij}$ avec $i, j \in W$

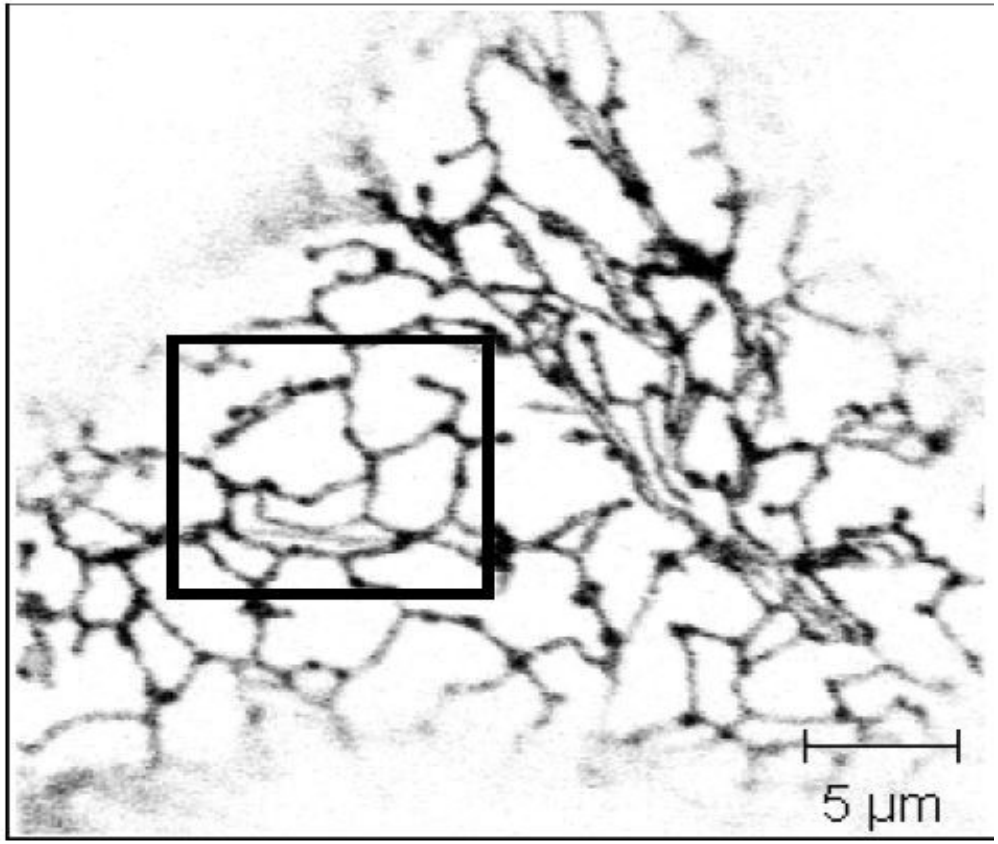
  - *Il ne peut pas y avoir trop d'aretes dans W*

⇒ ajouter la contrainte (1) ou (2) quand on a une sous-tournée dans une solution

TD programmation linéaire
ER: Reticulum endoplasmique

- Membrane bound organelle
- Protein translocation, vesicle budding *[grand central station for biochemical compounds]*

- A network of tubules [edges] connected by nodes
  - persistent ones (anchoring to plasma membrane)
  - non-persistent ones (tubule junctions and tubule ends sensible to remodelling)



E.g tobacco leaf © American Society of Plant Biologists. www.plantcell.org

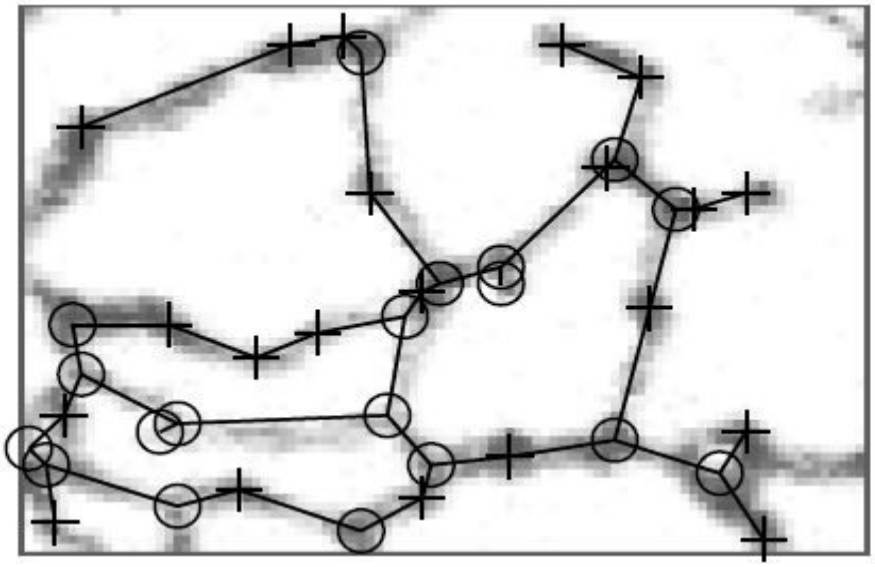
Understanding the ER dynamics

- Analysis of confocal microscopy data
- Is there and what is the optimization principle ?

TD programmation linéaire
ER: topologie des tubules

ER morphology in tobacco leaf epidermal cells

- Similar to Steiner trees or MST
- Additional cycles
- Branching nodes (3-way junctions) with angles around 120°



Abstracted geometric graphs

- From image processing :

O non-persistent nodes

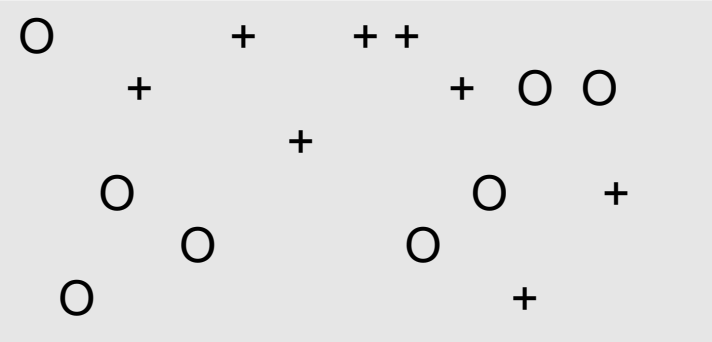
\+ persistent nodes

TD programmation linéaire
ER: modèle de base / complet

Compute a minimum length connected network

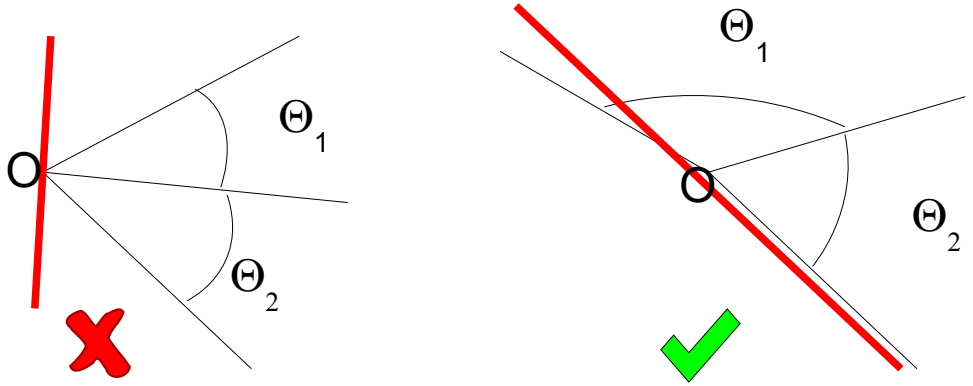
- Nodes (O/+) : tubule junctions extracted from images → V
- Detected branching nodes → $V_b \subseteq V$
- Plane graph

[basic model]



- Angle constraint on branching nodes

[full model]

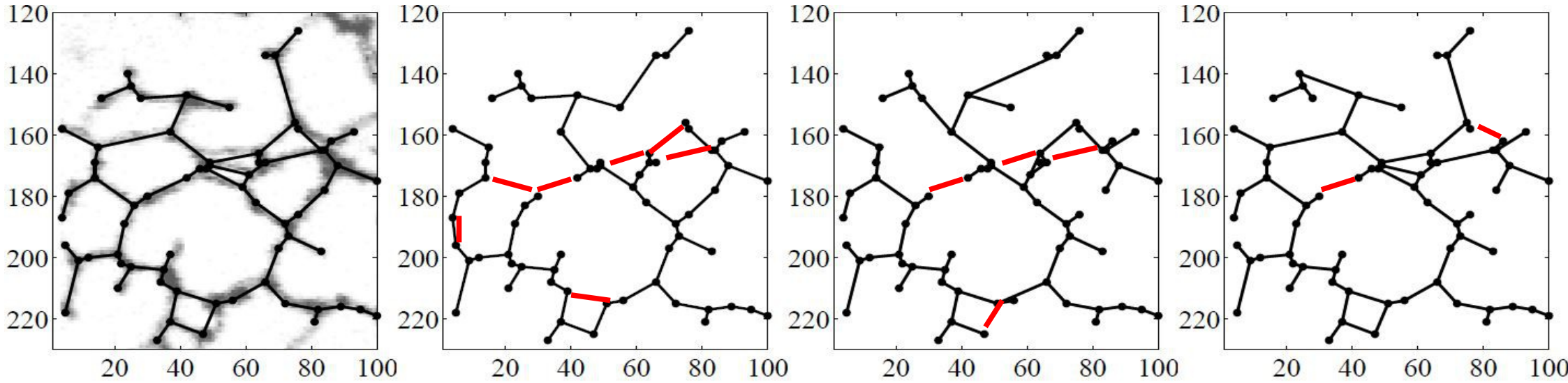


TD programmation linéaire
ER: contraintes initiales / additionnelles

- Modèle pour obtenir un réseau de taille minimale
  - Contraintes de degré sur les nœuds de V et V_b
  - Planarité
  - Connectivité
  - Géométrie

Computed network G_1

Abstracted network *G* | Min. Spanning Tree | Basic model | Full model



TD programmation linéaire
ER: contraintes initiales

- Modèle pour obtenir un réseau de taille minimale
- Modèle de base
  - Variables 0-1 X_{ij} : jonction ($i \leftrightarrow j$) sélectionnée ($i \in V, j \in V$)
  - Fonction objective ? Compter le nb de jonctions
  - Connectivité : de 1 à 3 connexions pour les sommets qui ne sont pas dans V_b et 3 connexions pour les sommets de V_b

Compter le nb de jonctions sommet par sommet

TD programmation linéaire
ER: contraintes initiales

- Modèle pour obtenir un réseau de taille minimale
- Modèle de base
  - Variables 0-1 X_{ij} : jonction ($i \leftrightarrow j$) sélectionnée ($i \in V, j \in V$)
  - Fonction objective ? *min* $\sum_{ij} X_{ij}$
  - Connectivité : de 1 à 3 connections pour les sommets qui ne sont pas dans V_b et 3 connexions pour les sommets de V_b

    $\forall i \in V \setminus V_b$, $1 \leq \sum_{i \neq j} X_{ij} \leq 3$

    $\forall i \in V_b$, $\sum_{i \neq j} X_{ij} = 3$

TD programmation linéaire
ER: contraintes aditionnelles

- A vérifier sur chaque solution :
- Connectivé : calcul des composante connexes

  Si on a une composante $W \subset V$, $2 \leq |W| < |V|$.

  Il faut une arête qui sort de W

- on rajoute des contraintes (cuts) et on recommence la résolution avec ces contraintes additionnelles jusqu'à ce que la solution soit complétement valide

TD programmation linéaire
ER: contraintes aditionnelles

- A vérifier sur chaque solution
- Connectivé : calcul des composante connexes

  Si on a une composante $W \subset V$, $2 \leq |W| < |V|$.

  $\delta(W) \geq 1$ avec $\delta(W) = \sum_{i,j} X_{ij}$ et $i \in W, j \notin W$

  - $\forall i \in V_b$, $\sum_{i \neq j} X_{ij} = 3$

TD programmation linéaire
ER: contraintes aditionnelles

- A vérifier sur chaque solution
- Si 2 arêtes (i ↔ j) et (k ↔ l) se croisent, il faut l'interdire

Exclusion mutuelle

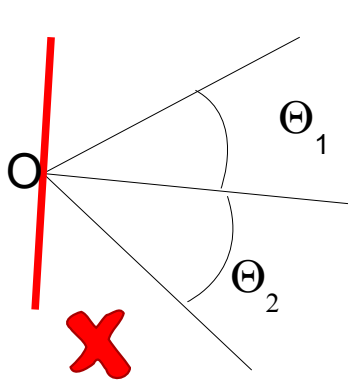
TD programmation linéaire
ER: contraintes aditionnelles

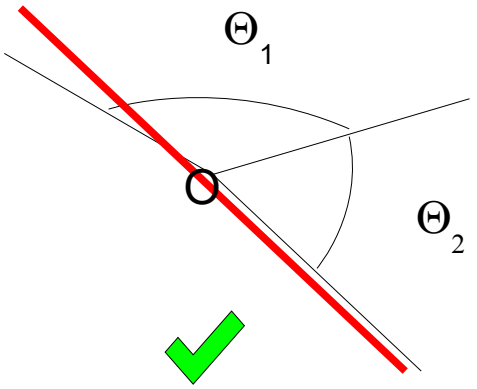
- A vérifier sur chaque solution
- Si 2 arêtes (i ↔ j) et (k ↔ l) se croisent, il faut l'interdire

$$X_{ij} + X_{kl} \leq 1$$

- Pour les nœuds de Vb, si les trois jonctions (i↔j, i↔k, i↔l) sont du même coté, il faut l'interdire

Exclusion d'1 sur les 3





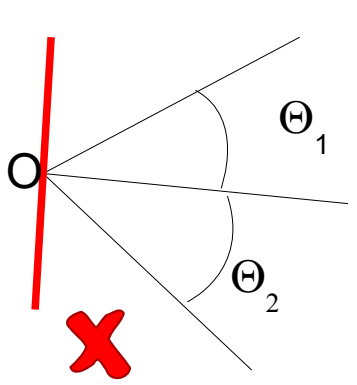
TD programmation linéaire
ER: contraintes aditionnelles

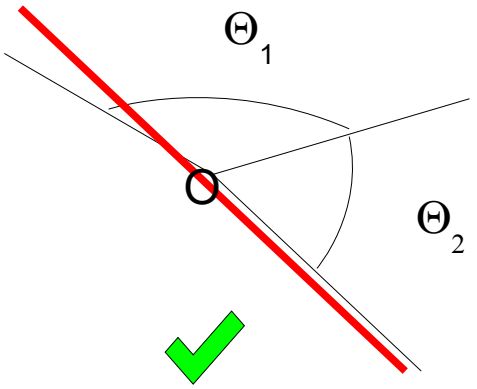
- A vérifier sur chaque solution
- Si 2 arêtes (i ↔ j) et (k ↔ l) se croisent, il faut l'interdire

$$X_{ij} + X_{kl} \leq 1$$

- Pour les nœuds de Vb, si les trois jonctions (i↔j, i↔k, i↔l) sont du même coté, il faut l'interdire

$$X_{ij} + X_{ik} + X_{il} \leq 2$$





Linear programming models

- Modeling techniques. How to translate model constraints and optimization criteria into linear formulas ?
- Some well-known optimization problems LP or IP formulation

Linear programming
Linear *formulettes*

- *For each unit of x_1 there must be at least x_2 units of x_2*

  $5x_1 \leq x_2$

- *A port can load either 11 x_1's per week, or 45 x_2's, or 30 x_3's. What combinations of x_1, x_2 and x_3 can be loaded in 10 weeks ?*

  $(1/11)\ x_1 + (1/45)\ x_2 + (1/30)\ x_3 \leq 10$

- *Process x_1 produces 24.5 units of x_2 and 73.1 units of x_3 per hour*

  $x_2 = 24.5\ x_1 \quad x_3 = 73.1\ x_1$

- *Sorties of type x_1 must constitute at most 33% of all sorties of type x_1, x_2, and x_3*

  $0.67\ x_1 \leq 0.33\ x_2 + 0.33\ x_3$ $\qquad \dfrac{x_1}{x_1 + x_2 + x_3} \leqslant \dfrac{33}{100}$

Linear programming
Logic *formulettes* with 0-1 variables

- *z_1 only if z_2, z_1 is sufficient for z_2, z_1 implies z_2*

  $z_1 \le z_2$
- *z_3 implies z_1 and z_2*

  $z_3 \le z_1 \quad z_3 \le z_2 \quad z_3+1 \ge z_1 + z_2$
- *z_3 implies z_1 or z_2 (at least one, inclusive or)*

  $z_3 \ge z_1 \quad z_3 \ge z_2 \quad z_3 \le z_1 + z_2$
- *z_3 implies z_1 xor z_2 (at most one, exclusive or)*

  $z_3 \le z_1+z_2 \quad z_3 \ge +z_1-z_2 \quad z_3 \ge -z_1+z_2 \quad z_3 \le 2-z_1-z_2$
- *Not z*

  $1 - z$

G. Brown

Linear programming
Set *formulettes* with 0-1 variables

- *From set $z_1, z_2, \ldots z_k$, select at least one (covering)*

  $z_1 + z_2 + \ldots + z_k \geq 1$
- *From set $z_1, z_2, \ldots z_k$, select at most one (packing)*

  $z_1 + z_2 + \ldots + z_k \leq 1$
- *From set $z_1, z_2, \ldots z_k$, select exactly one (partitioning)*

  $z_1 + z_2 + \ldots + z_k = 1$
- *From set $z_1, z_2, \ldots z_k$, select more than 3 (cardinality)*

  $z_1 + z_2 + \ldots + z_k \geq 4$
- ...

G. Brown

Linear programming
Quadratic formula

Linearize quadratic constraint

- *z_1 and z_2 is true*

$z_1.z_2 \rightarrow y \Leftrightarrow z_1.z_2$

$y \leq z_1$

$y \leq z_2$

$z_1 + z_2 - 1 \leq y$

Bin packing problem
IP formulation

- n boxes of capacity C
- n items of respective size c_1, c_2, …, c_n

  x_{ij} = 1 means item i stored into box j, 0 otherwise
- Constraints : each item is stored in one box, and boxes cannot be overfilled
- Objective function : minimize the number of boxes used

$$\forall \text{item } i \text{ and box } j: \quad x_{ij}, y_j \in \mathbb{N}, 0 \leqslant x_{ij}, y_j \leqslant 1$$

$$\forall \text{box } j : \sum_{items} c_i x_i \leqslant C y_j$$

$$\forall \text{item } i : \sum_{boxes} x_{ij} = 1$$

$$\min \sum_{\text{boxes}} y_i$$

Knapsack problem
IP formulation

- One knapsack of capacity C
- n items of weight w_1, w_2, …, w_n and price (benefit) p_1, p_2, …, p_n .

  x_{ij} = 1 means item i is selected, 0 otherwise
- Constraints : the total weight of selected items less than C.

- objective function : maximize the benefit of knapsack

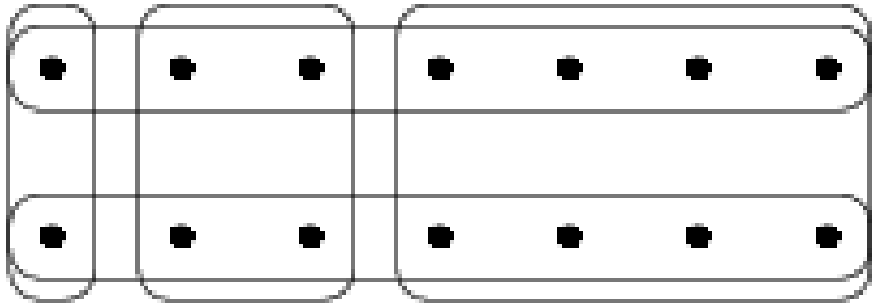
$$\forall \text{item } i \ : \quad x_i \in \mathbb{N}, 0 \leqslant x_i \leqslant 1$$

$$\sum_{items} w_i x_i \quad \leqslant C$$

$$\max \sum_{\text{items}} p_i x_i$$

Modeling
Antenna covering problem

- Antenna location choice
  - Set covering problem

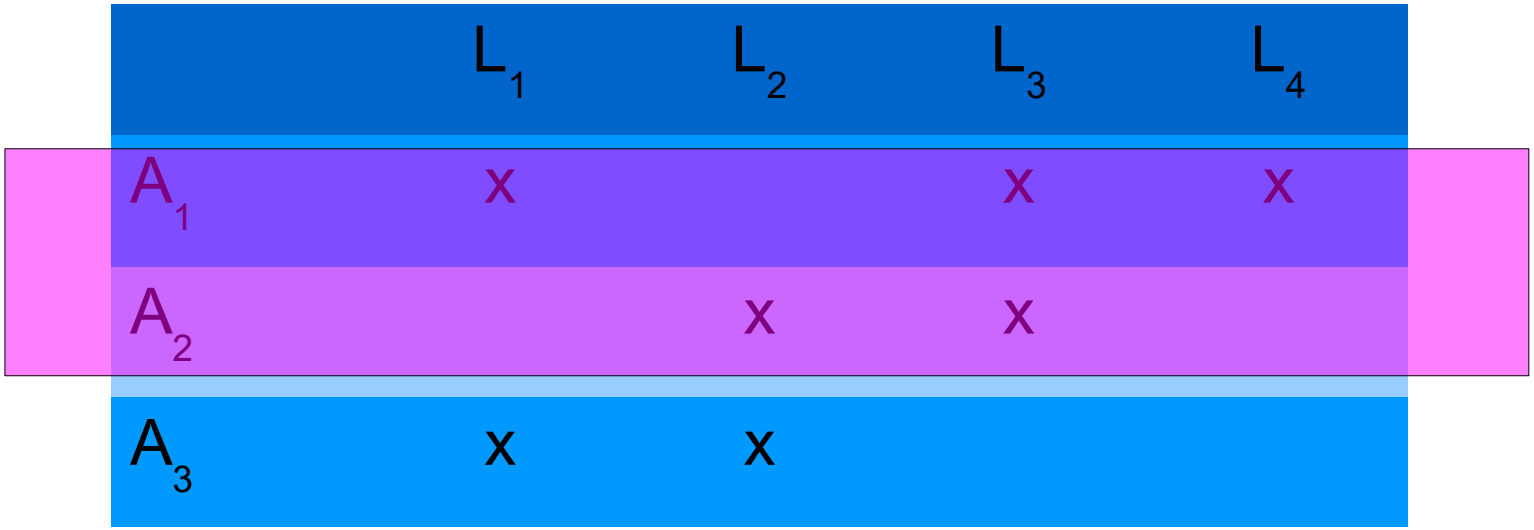


- Covering matrix
  - Is Location L_j covered by antenna A_i ?

	L_1	L_2	L_3	L_4
A_1	x		x	x
A_2		x	x	
A_3	x	x		

Modeling
Antenna covering problem

- Covering matrix
  - Is Location L_j covered by antenna A_i ?



- Find a minimal subset of potential antenna that covers all of the locations (maybe costs c_i)

Antena covering problem
IP formulation

- Variables : one binary variable x_i per antenna

  $x_i = 1$ means A_i is chosen, 0 otherwise
- Constraints : each location must be covered by at least 1 antenna

- objective function : minimize the number/cost of chosen subset of antenna

$$\forall \text{antenna} \quad i \quad : \quad x_i \in \mathbb{N}, 0 \leqslant x_i \leqslant 1$$

$$\forall \text{location} \quad j \quad : \quad \sum_{i \, covers \, j} x_i \quad \geqslant 1$$

$$\min \sum_{\text{antennas}} x_i \qquad \text{or} \min \sum_{\text{antennas}} c_i x_i$$

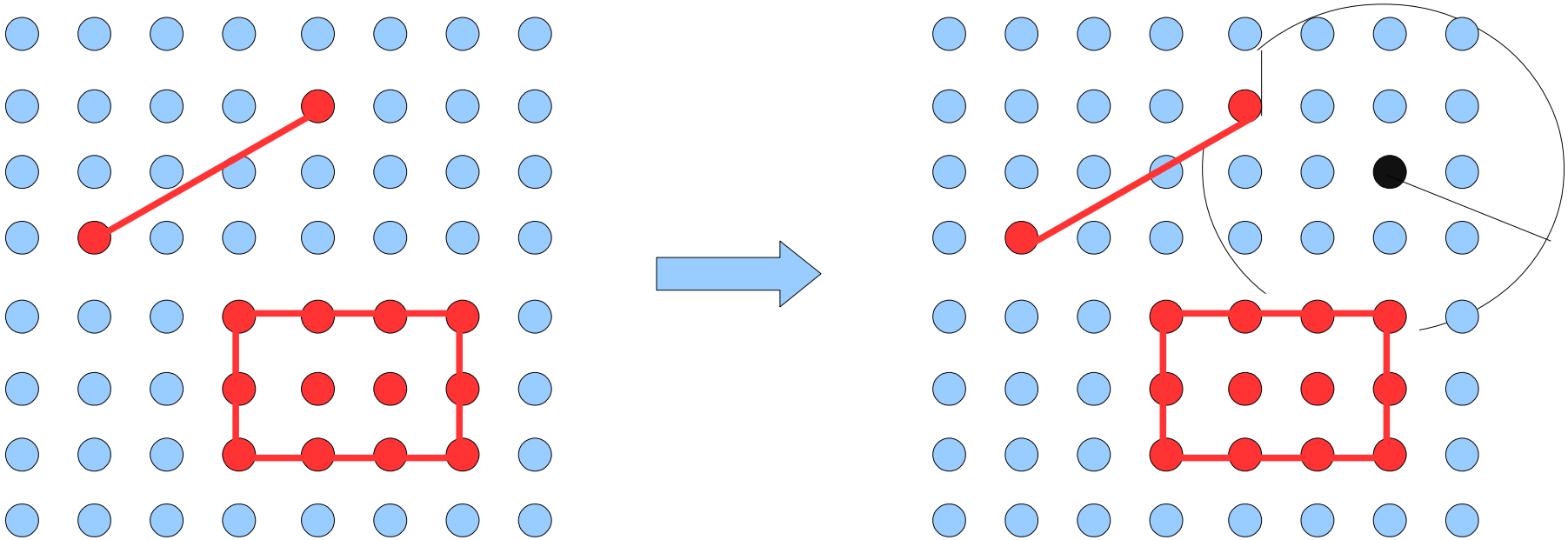
Antena covering problem
lab

- Covering an area with GSM capabilities
  - Placing antenas (antena range and cost) at lowest cost
  - Points to be covered are a superset of possible locations
    - To be covered by an antena, a point must lie into its range
    - It must be visible (see below)

- Possible constraints : define obstacles and non covered areas
- Degree of freedom on antena types ?

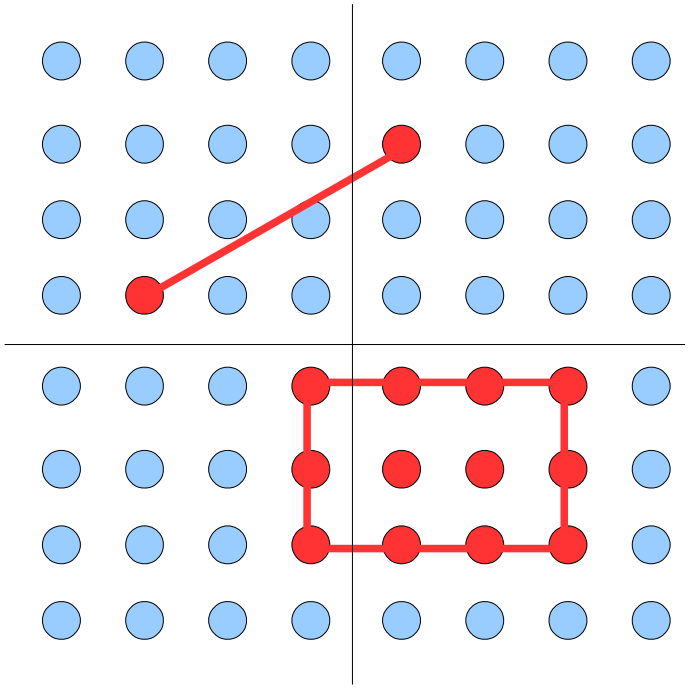
Antena covering problem
lab

Locate an antena, and check which points are covered



Antena covering problem
lab

IP formulation [optional : + Divide and Conquer DP method]



Dynamic programming algorithm

- Split the area
- Solve subproblems
- Aggregate solutions
- Remove useless antena by solving a new covering problem restricted to found locations