Modélisation par graphes de données volumiques médicales et applications à la correction topologique

Quentin Fortier
Encadrants: Jean-Marie Favreau, Vincent Barra et Marco Attene

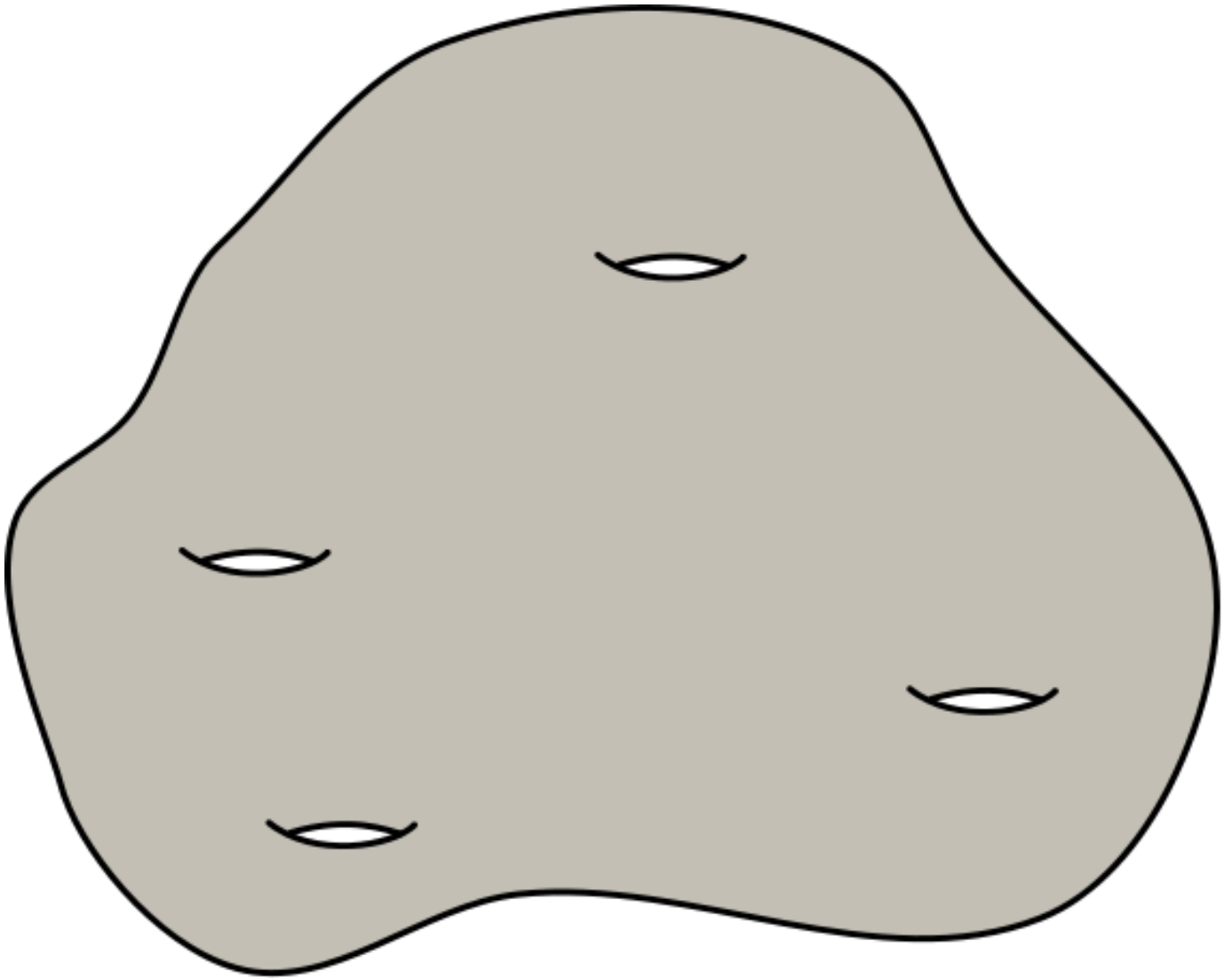
LIMOS

9 Septembre 2010

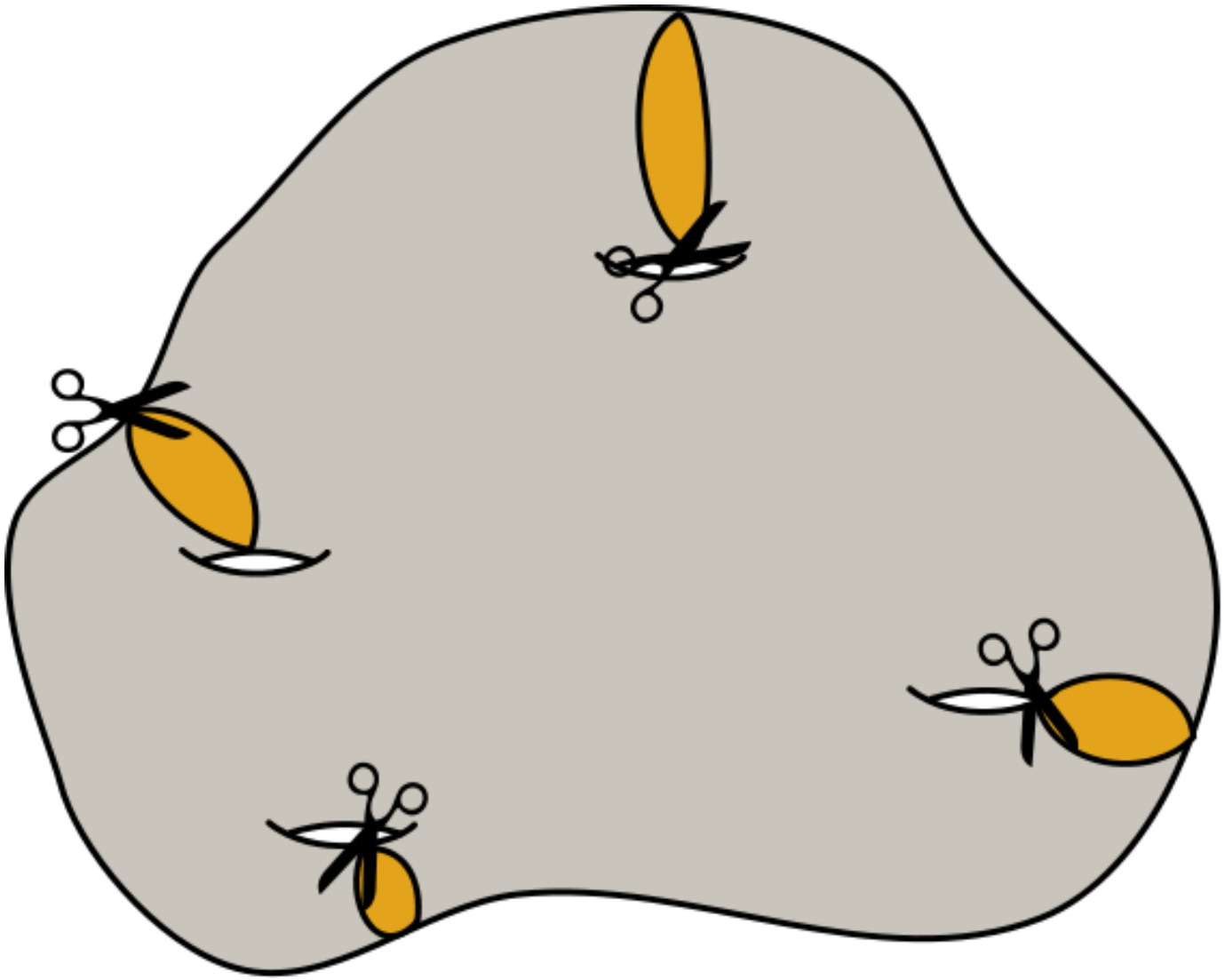
ENS DE LYON

Volume

Donnée: volume (connexe) de genre g (avec g "poignées" ou "trous")



Le but est de découper ce volume de façon à enlever les trous (en conservant la connexité) et de façon optimale

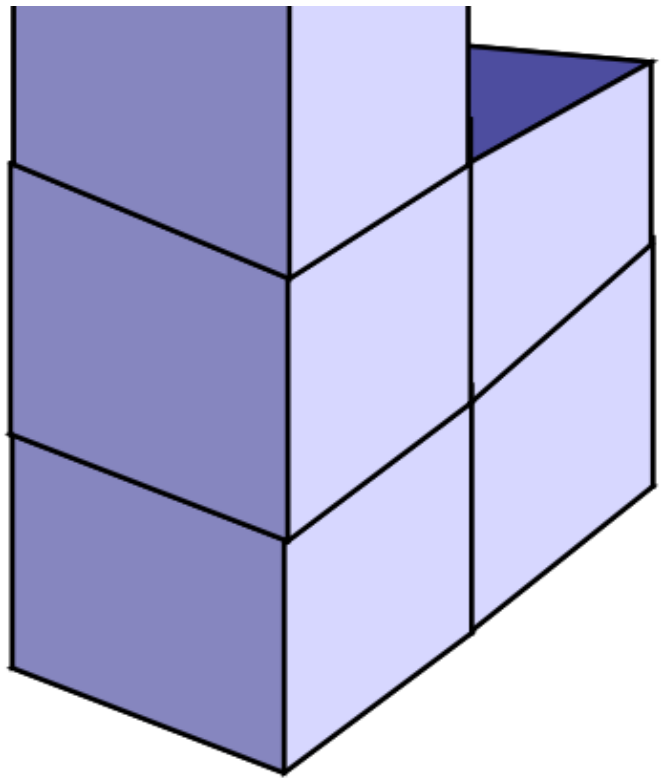


Volume discretisé

Algorithmiquement, un volume est représenté comme un ensemble de tétraèdres ou de voxels (cubes)

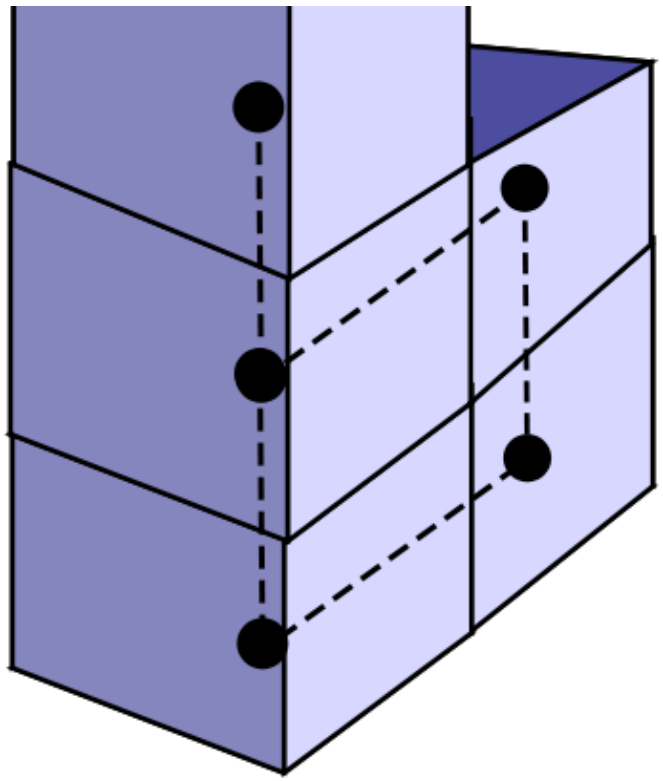
Dual

On construit alors le graphe dual:



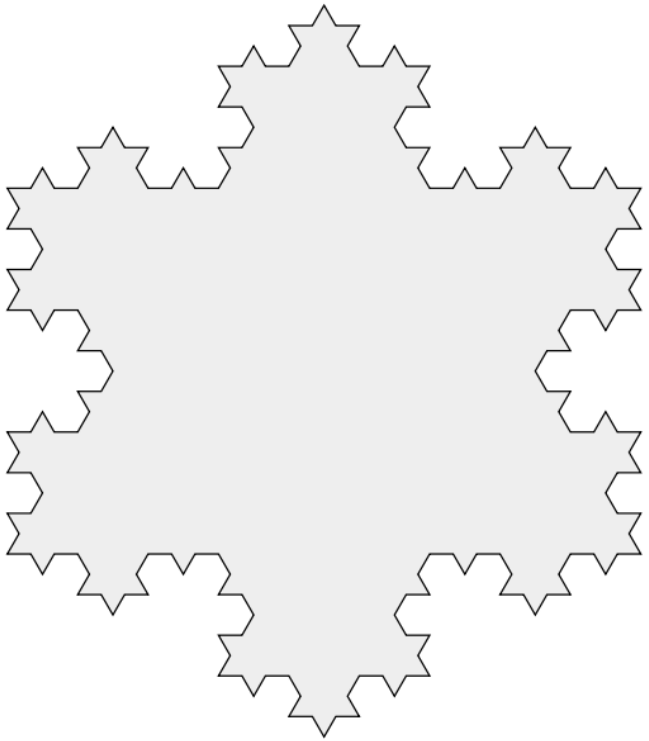
Dual

On construit alors le graphe dual:



Découpage de surface

Attention, minimiser l'aire d'une surface ne revient pas à minimiser son périmètre!



Etapes

- Trouver un premier découpage, pas forcément "bon"...

Etapes

- Trouver un premier découpage, pas forcément "bon"...
- Optimiser localement chaque cut tant que possible

Etapes

- Trouver un premier découpage, pas forcément "bon"...
- Optimiser localement chaque cut tant que possible
- Extraire des cuts résultant un découpage en un volume voulu (connexe de genre 0)

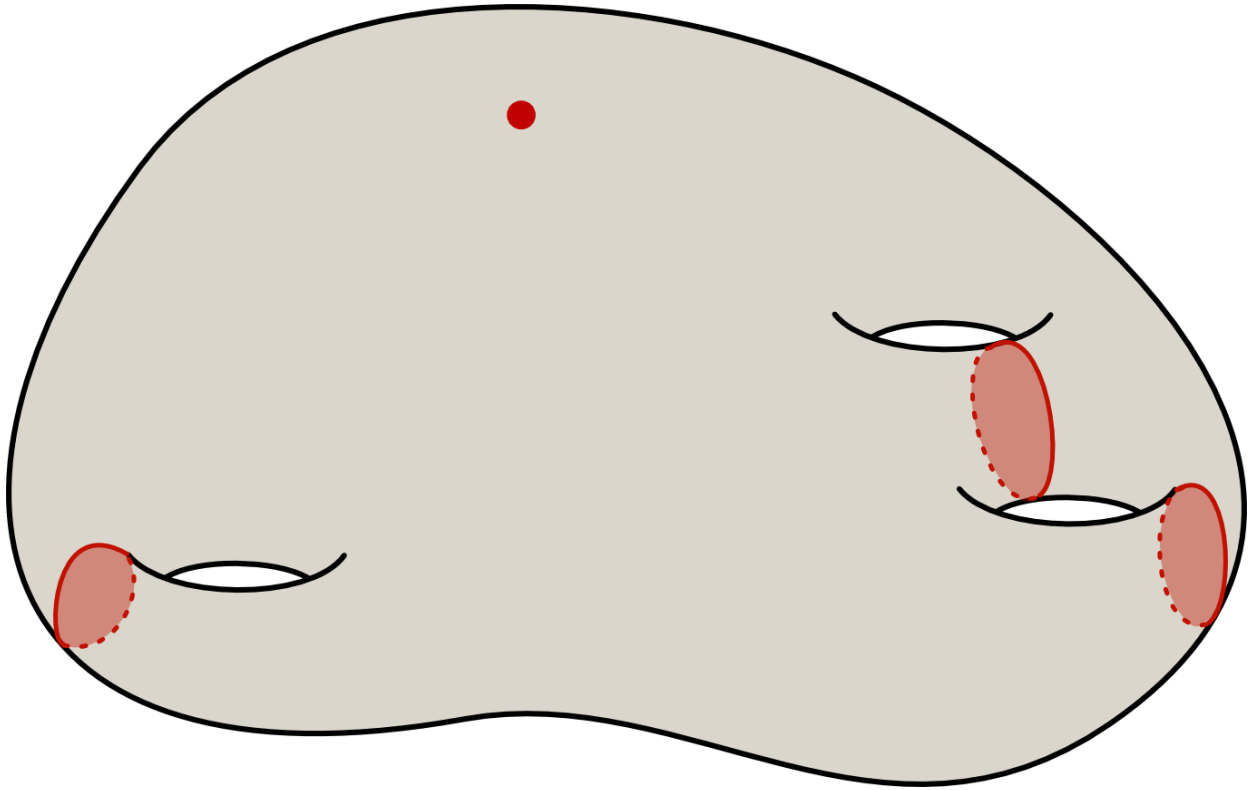
Cut locus

Pour construire un premier découpage, on utilise un cut locus

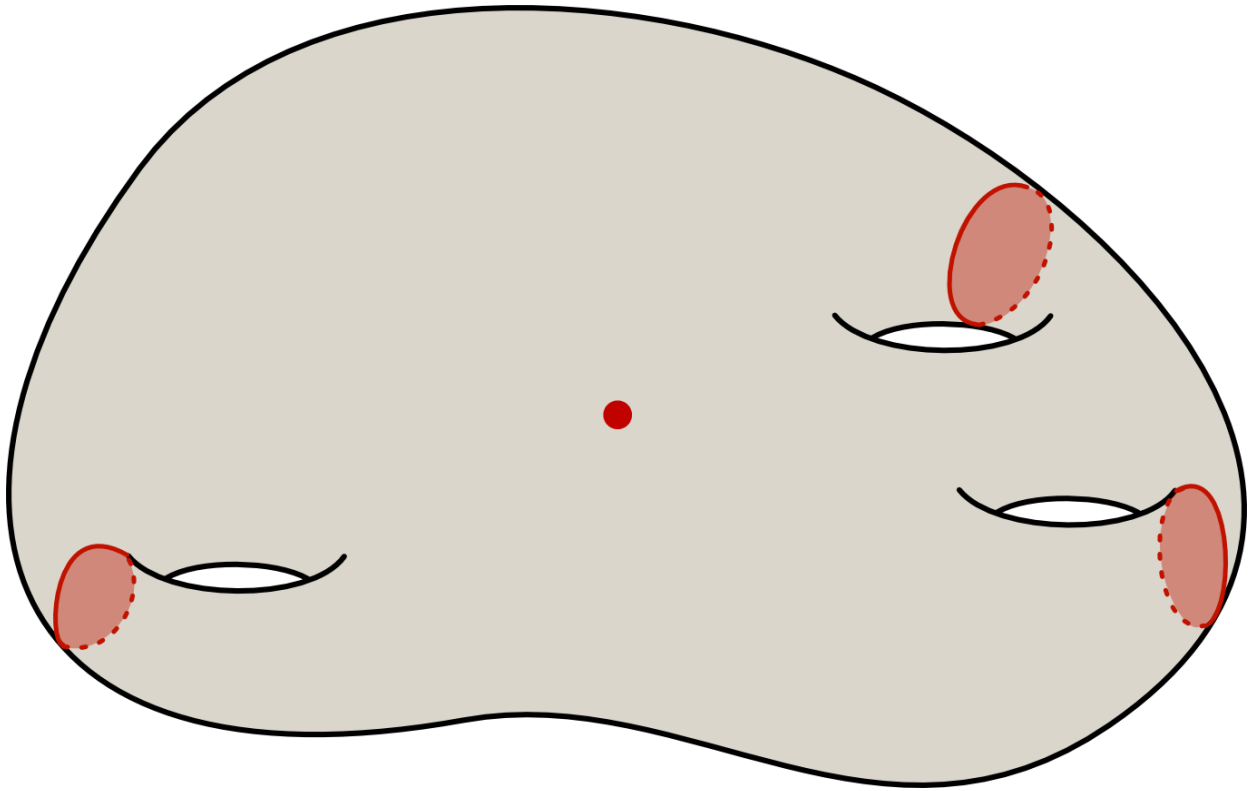
Cut locus

Le A-cut locus est l'ensemble des points p tels qu'il existe au moins deux plus court chemins de A à p

Pour A réduit à un point p, on obtient une seule composante connexe:

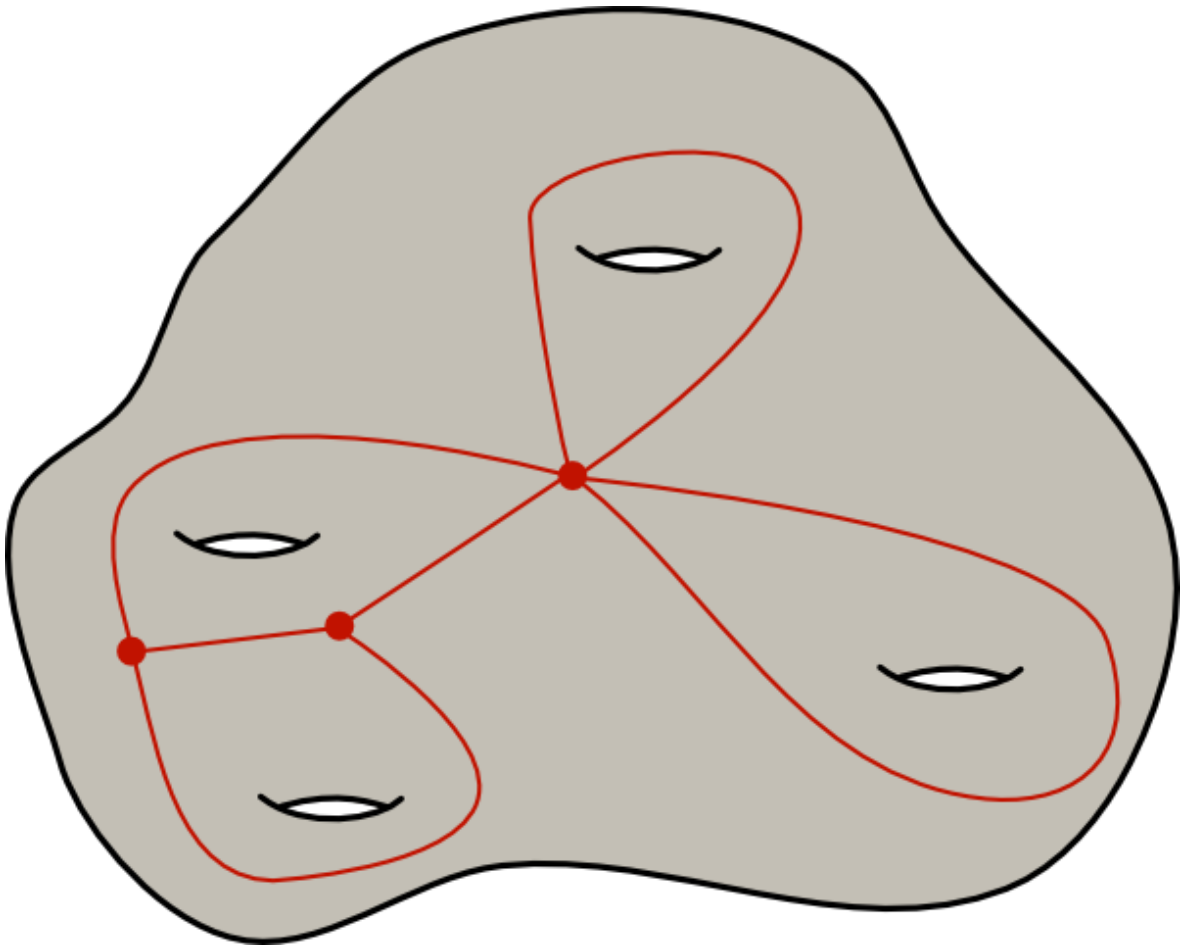


Le cut locus dépend bien sûr du point choisi:

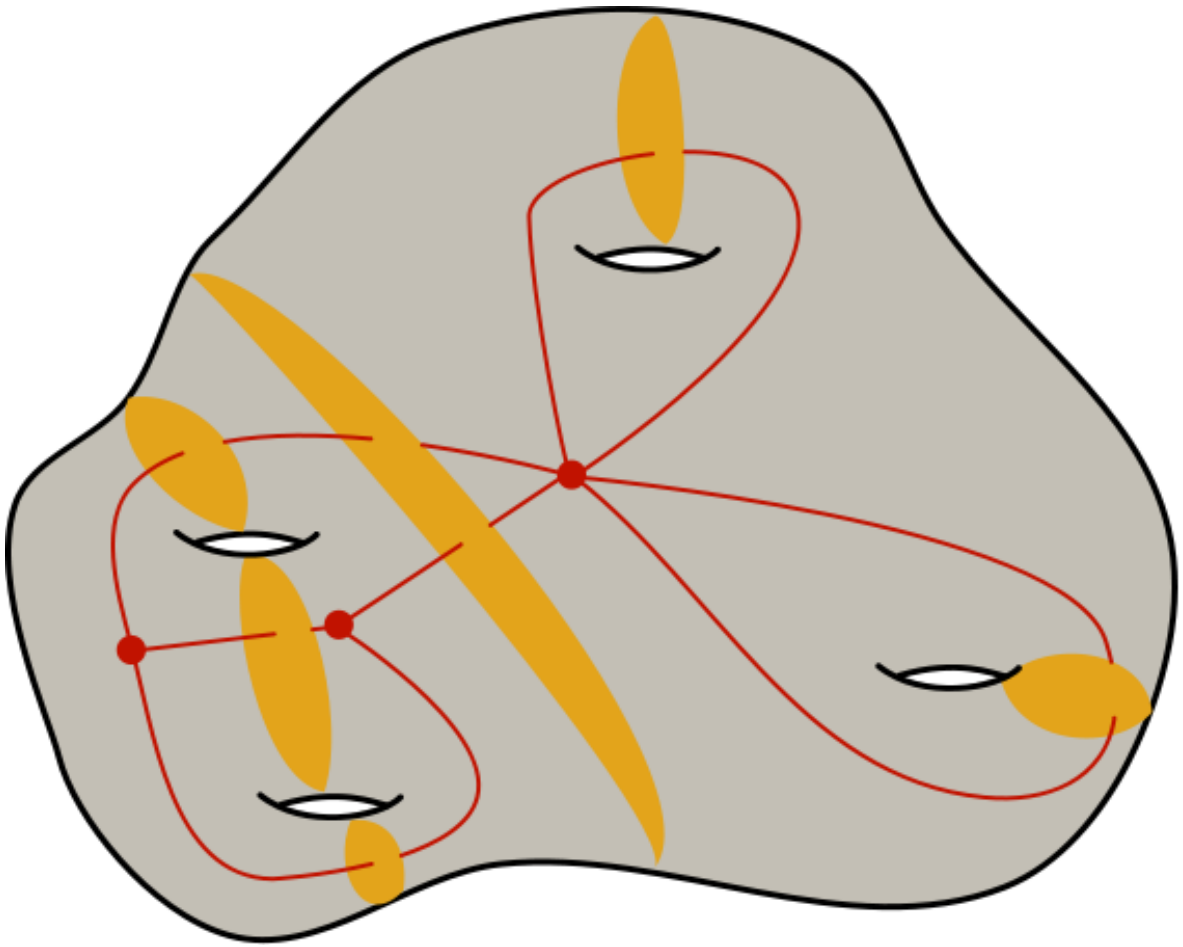


Squelette

On peut aussi prendre pour A les points d'intersection du squelette du volume:

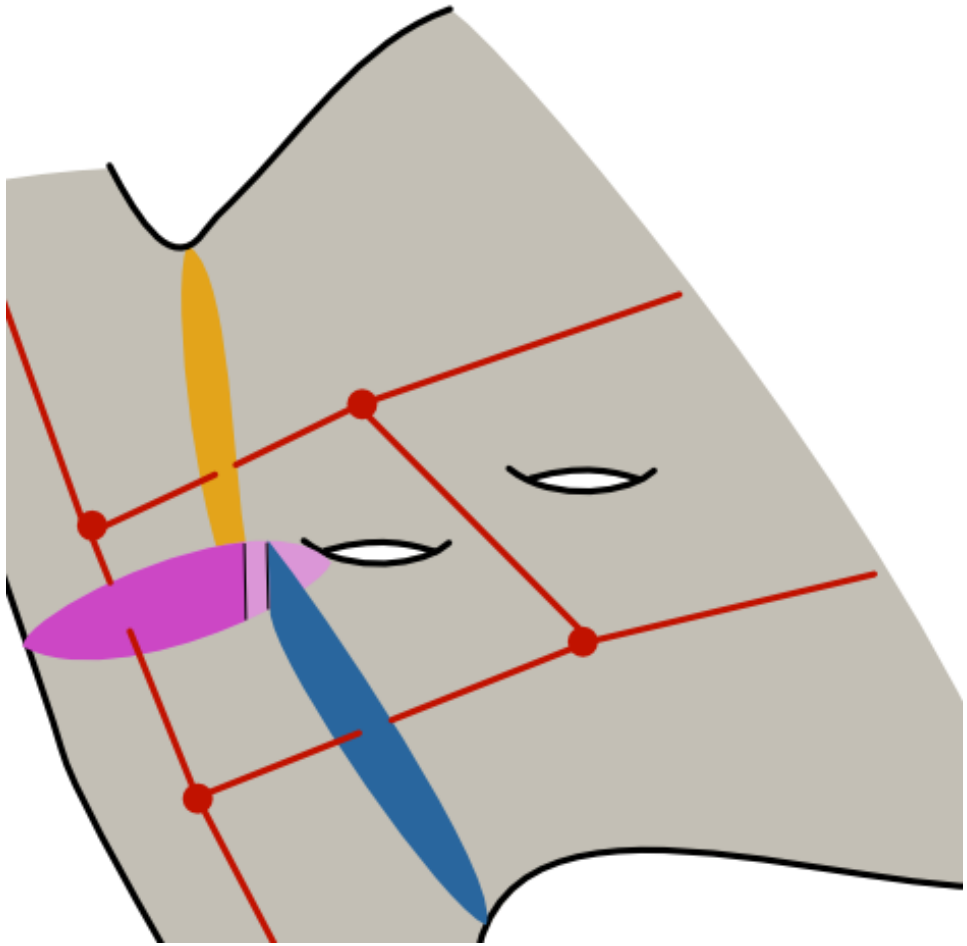


On peut aussi prendre pour A les points d'intersection du squelette du volume:



Intersections

Mais on peut avoir des intersections entre cuts!

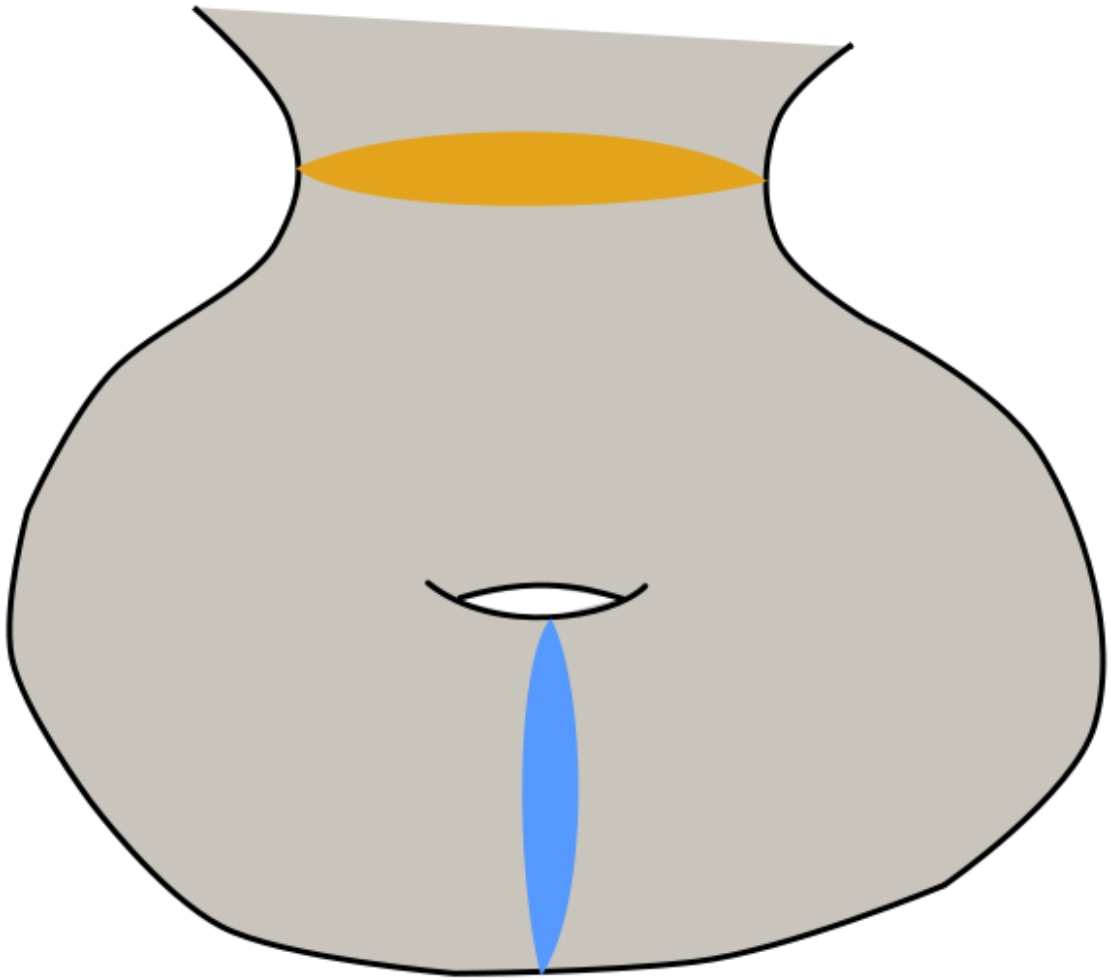


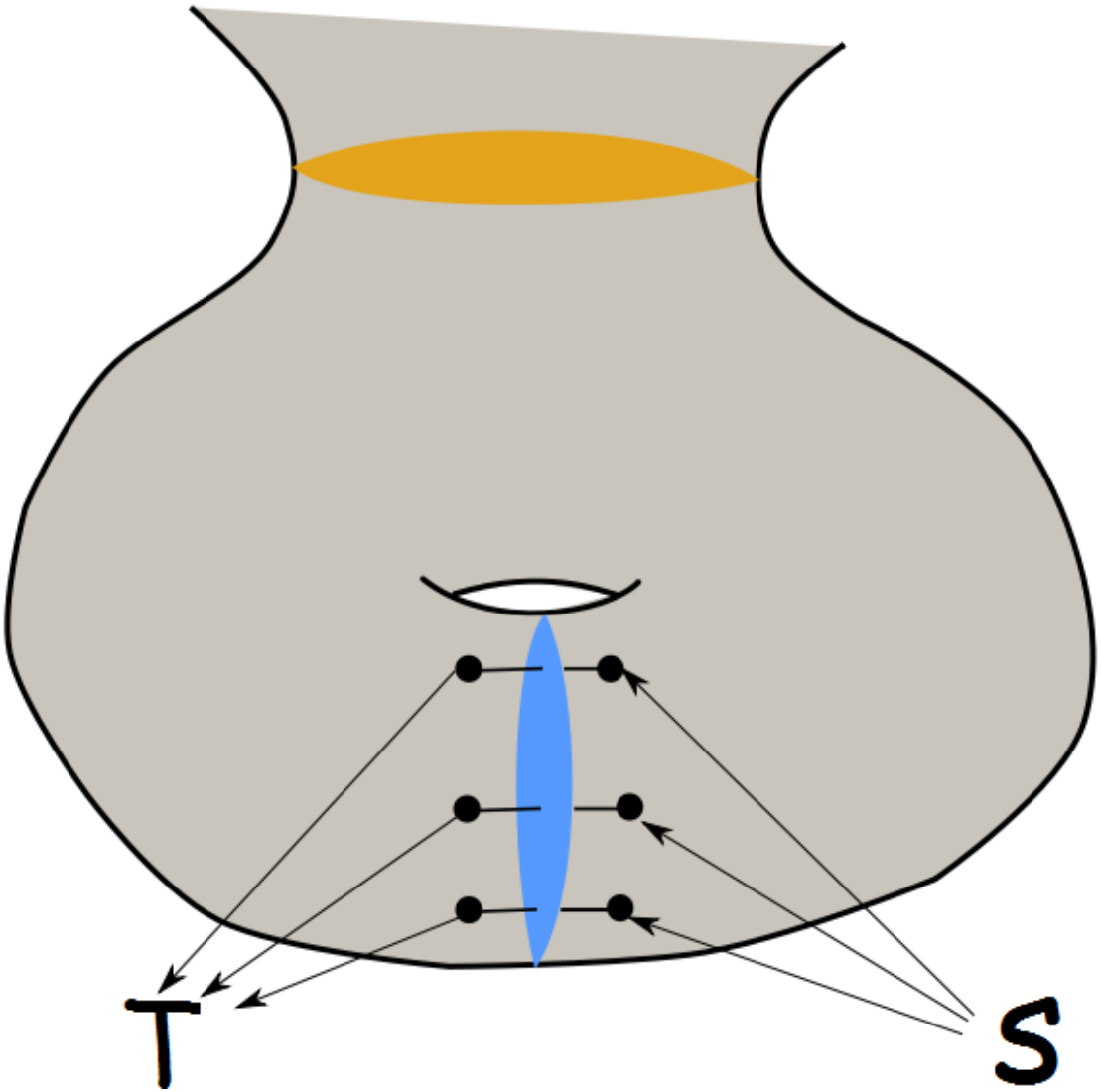
Comment optimiser un cut?

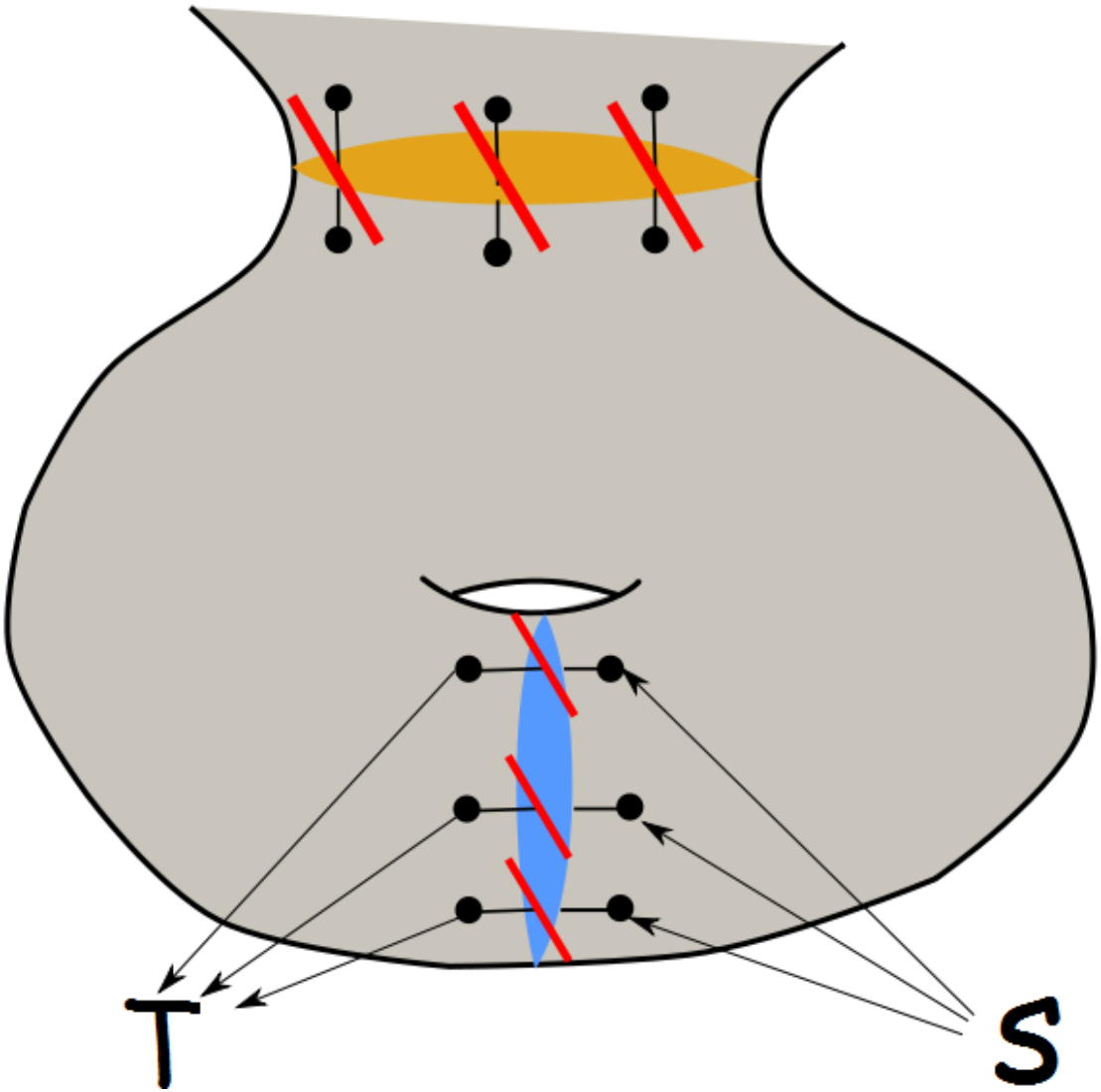
Comment optimiser un cut?
Utiliser le théorème max flow - min cut sur le dual!

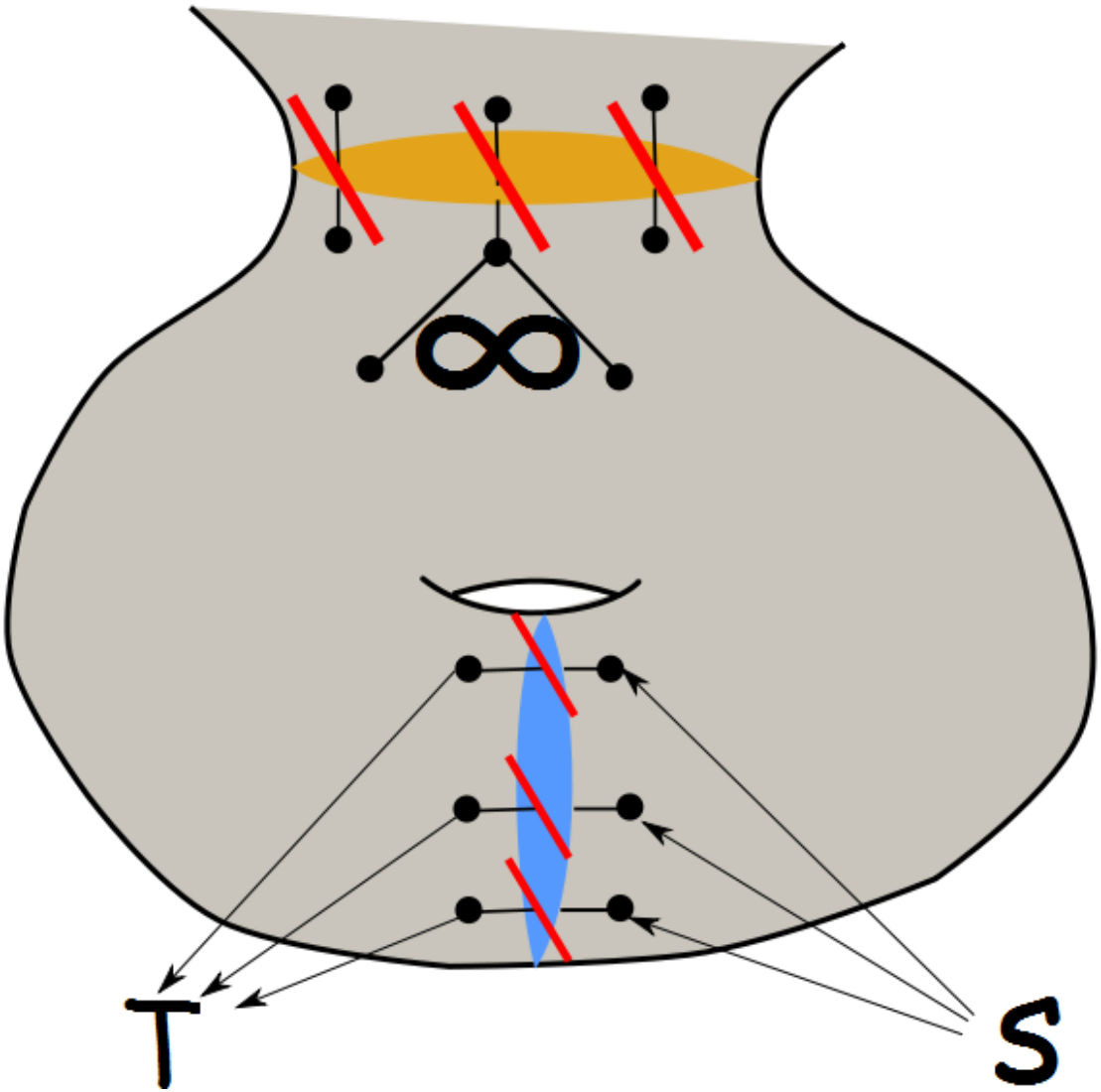
Auto cut

Un cut est auto si les deux "côtés" du cut sont dans la même composante connexe du volume découpé



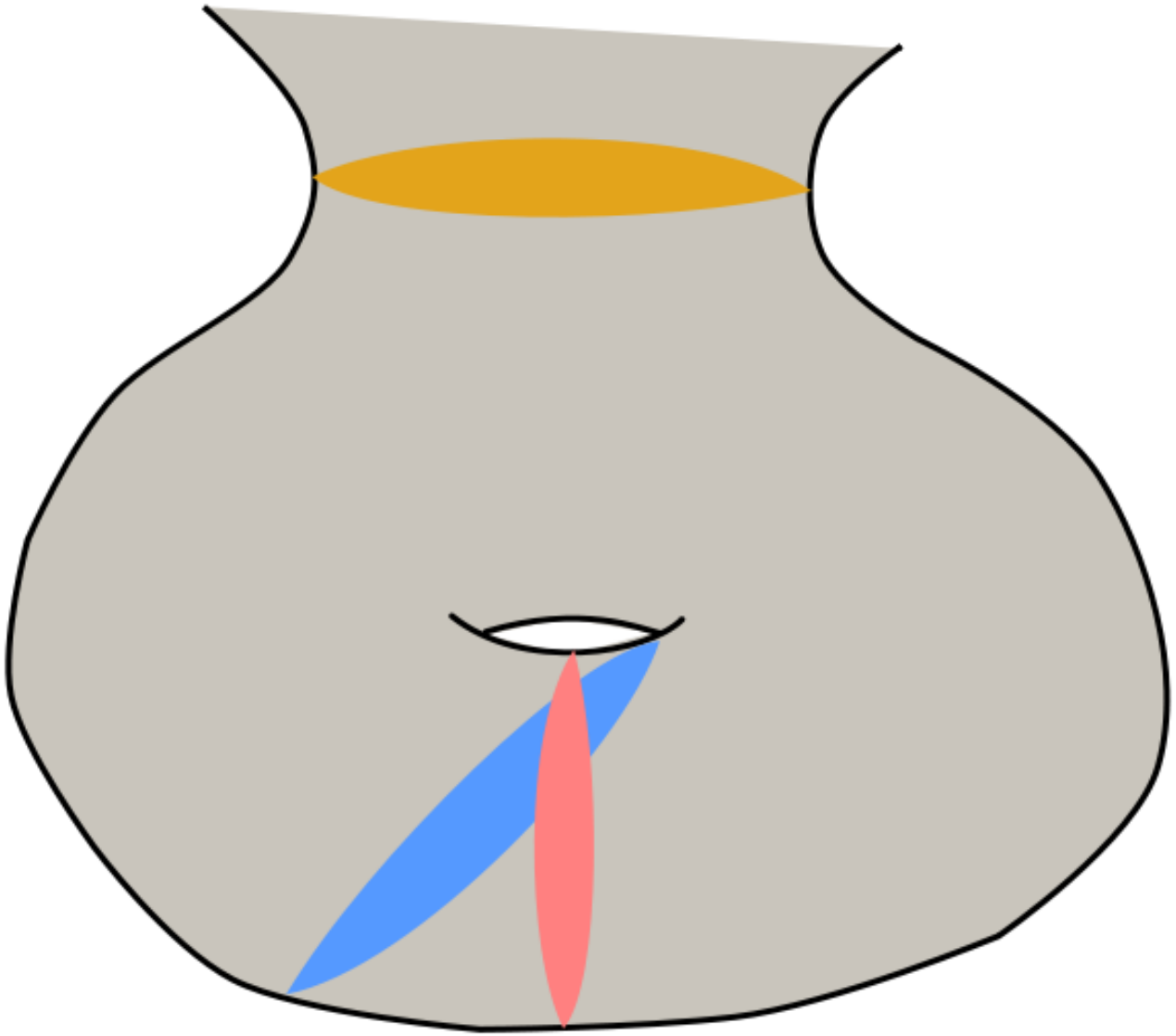

T
S


T
S

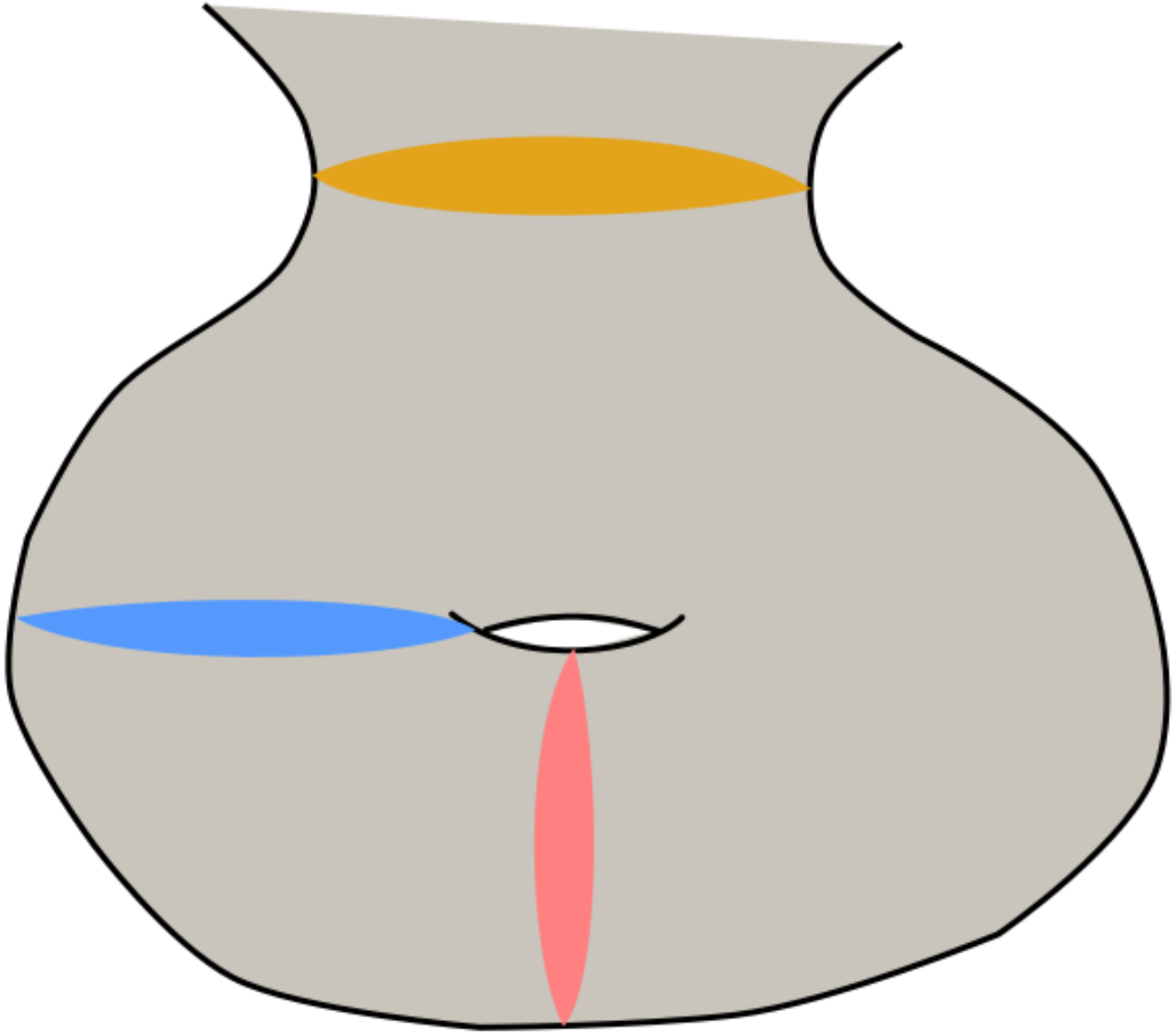

∞
T
S

Et puis on cherche un max flow (et un min cut)

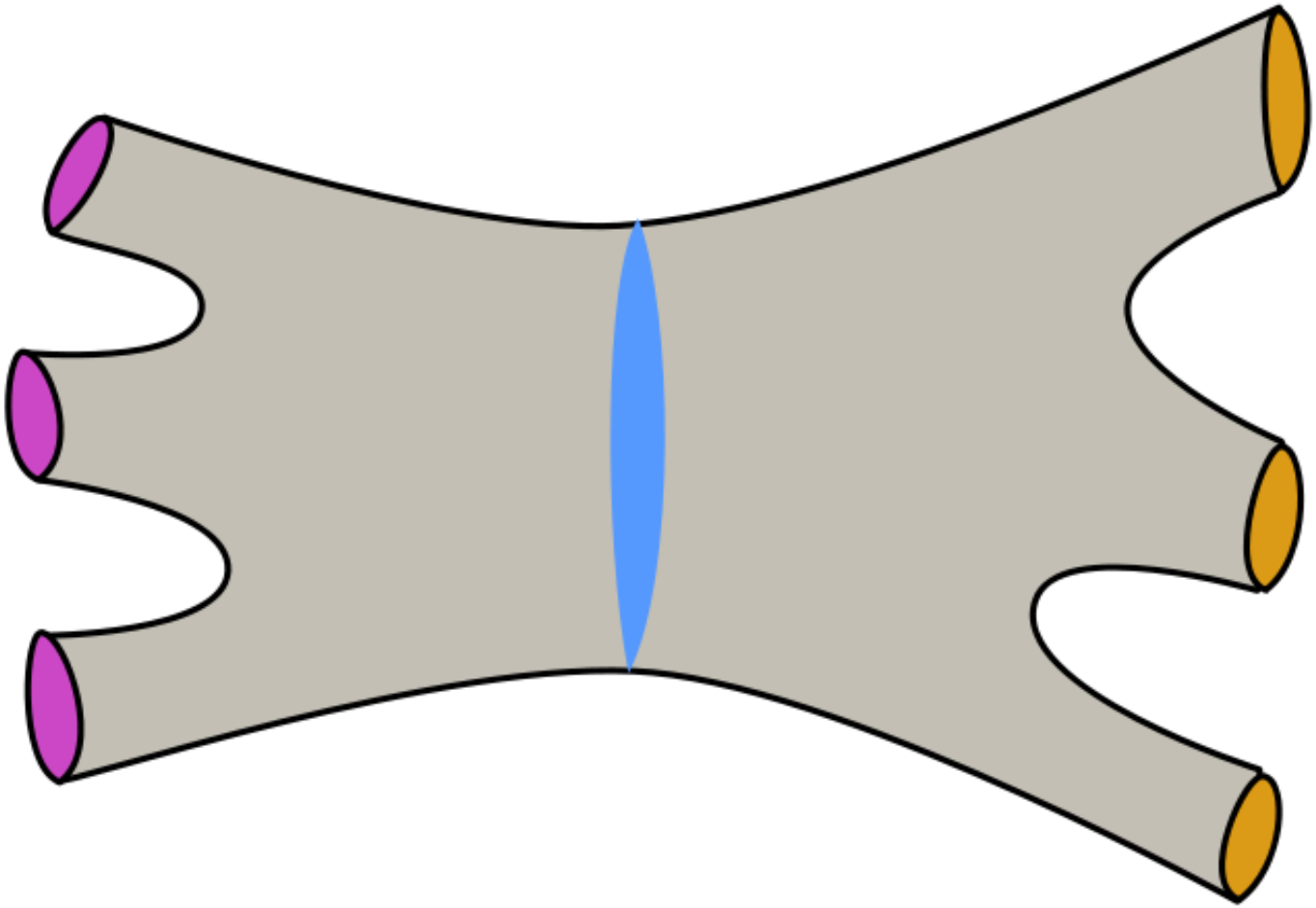
On peut avoir quelques cas particuliers...



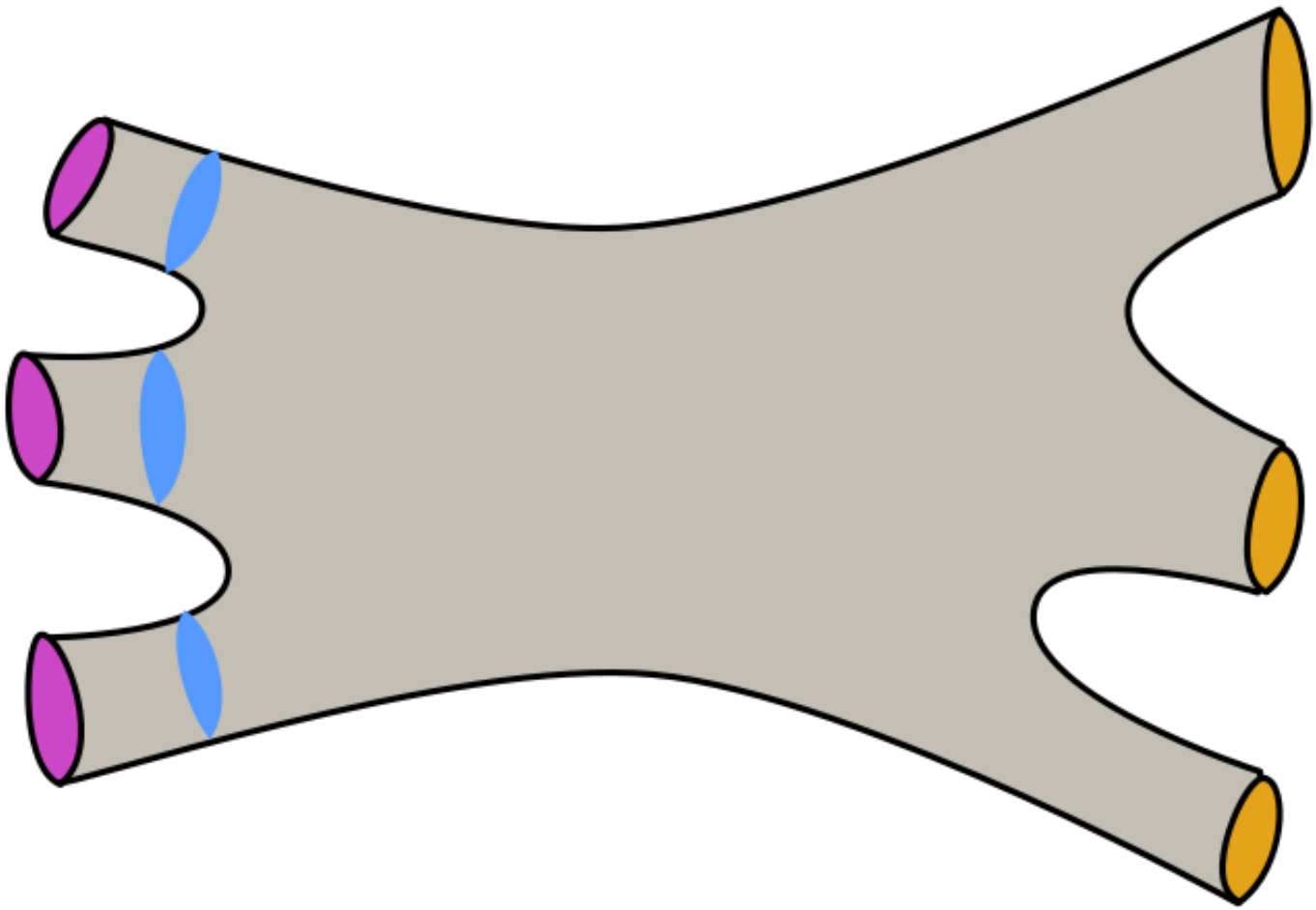
Solution: faire deux max flow successifs



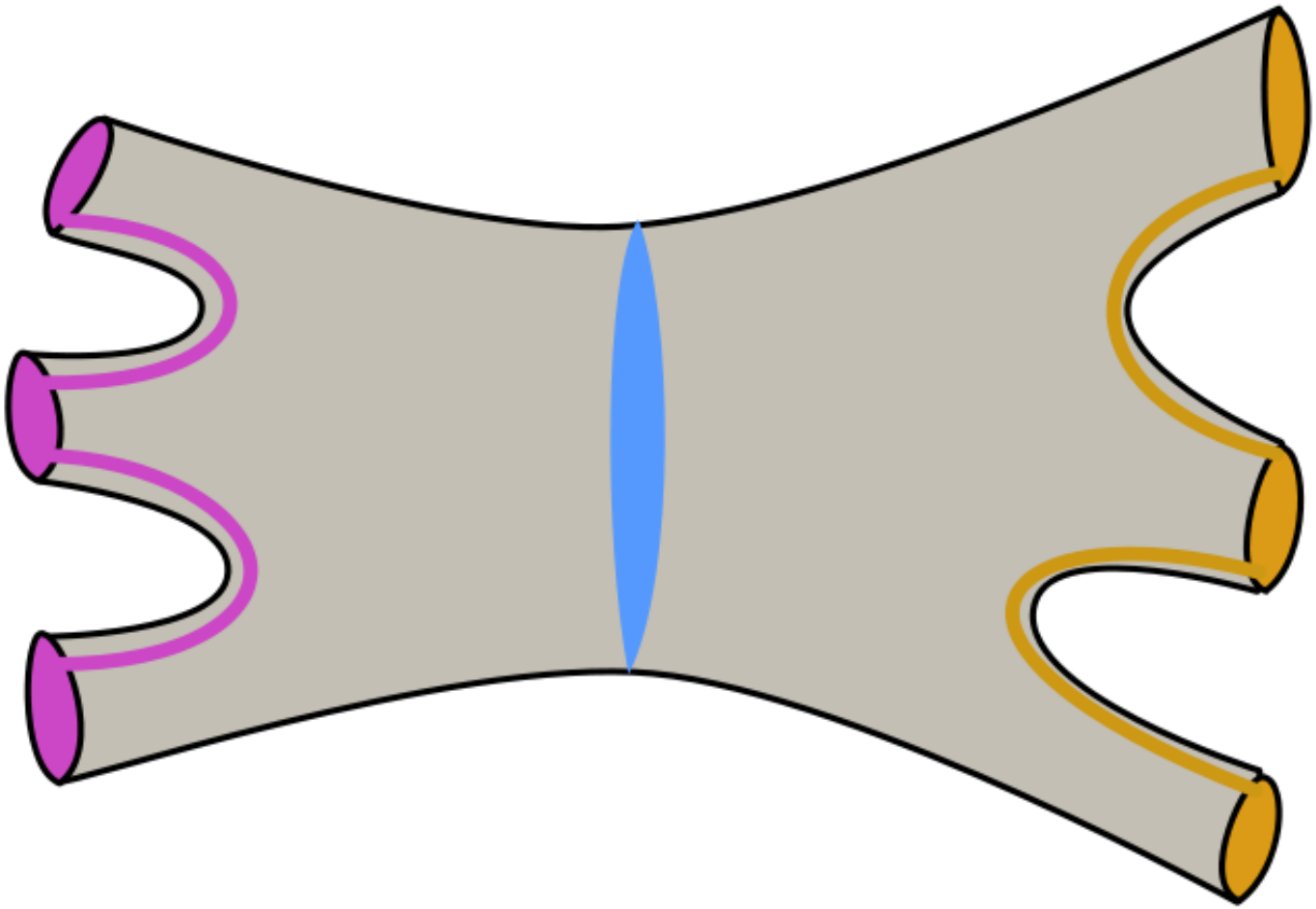
Cut "normal"

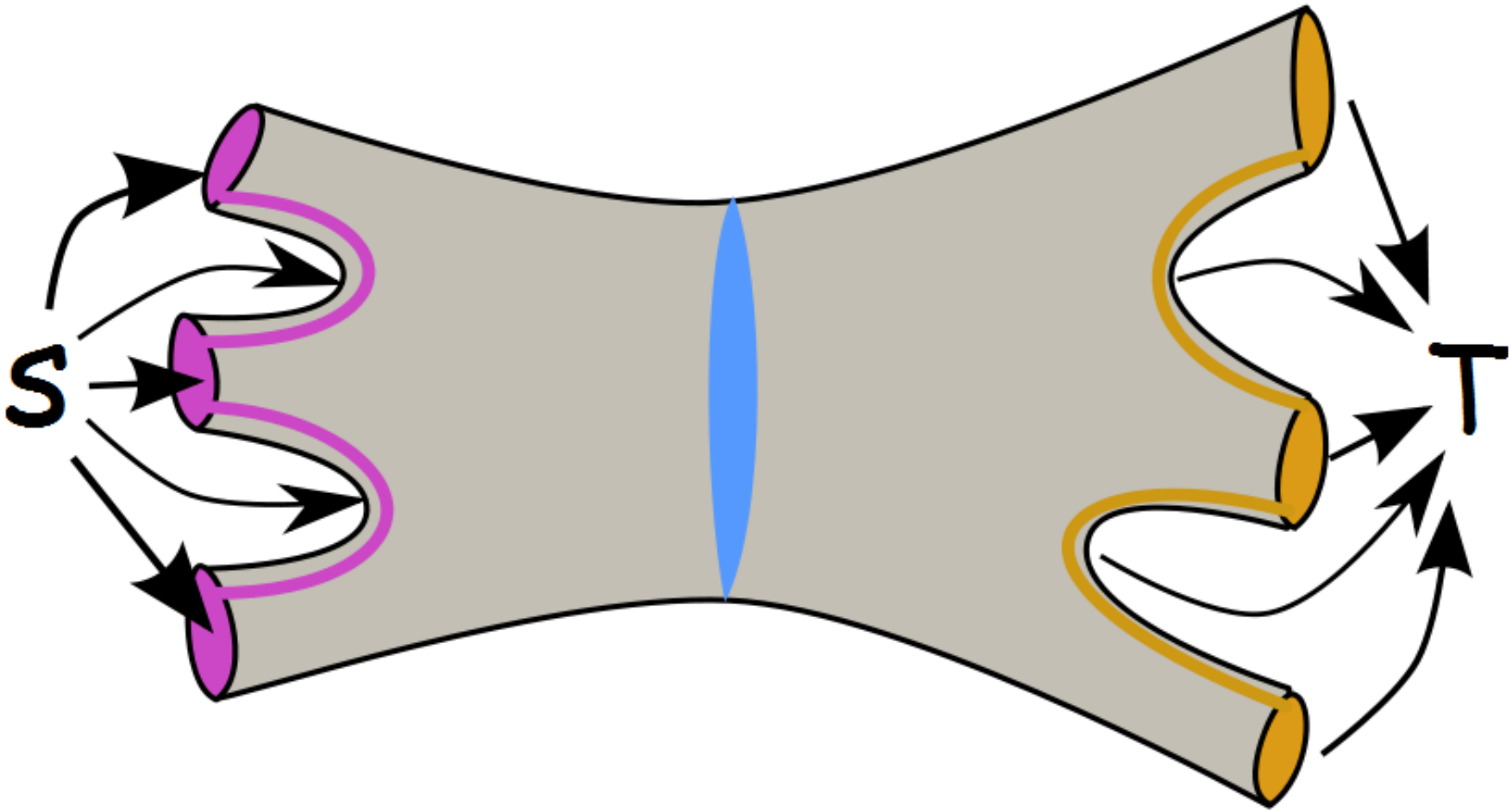


On préfère éviter d'obtenir un min cut non connexe

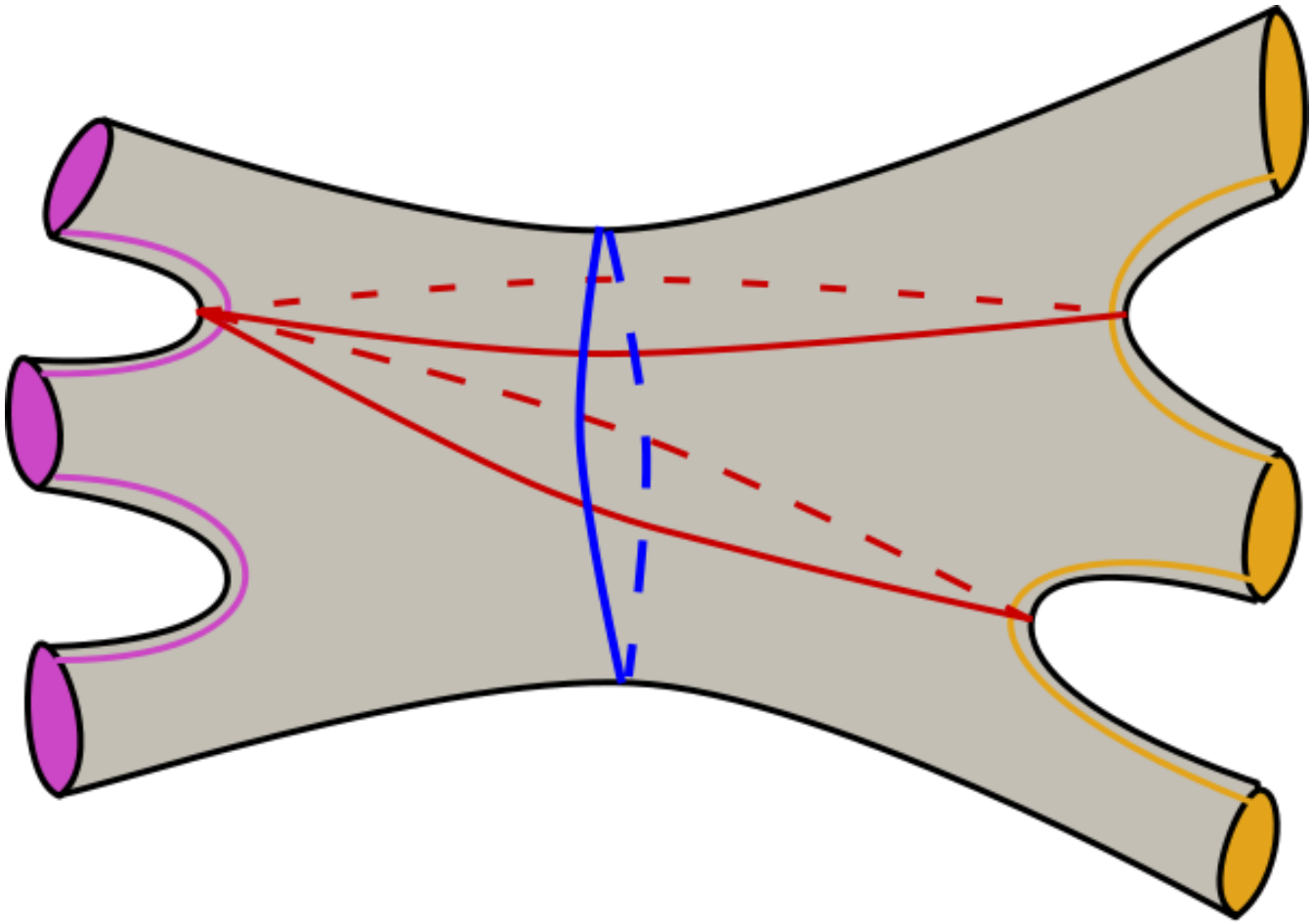


Pour cela on découpe selon un arbre de poids minimal connectant chaque cut (NP difficile), et ceci de chaque côté




S
T

On "oublie" plus de min cut potentiels quand on optimise un cut "normal"



Choix des cuts à optimiser

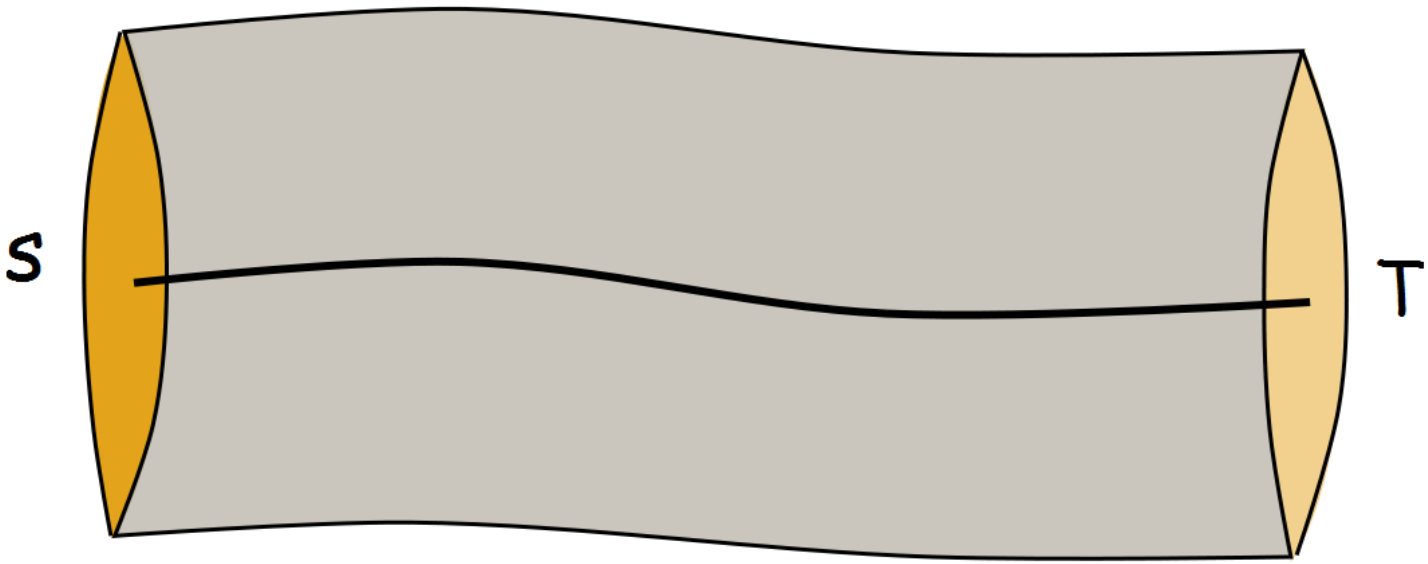
Malheureusement on ne connais pas de "meilleur" méthode pour sélectionner les cuts à optimiser:

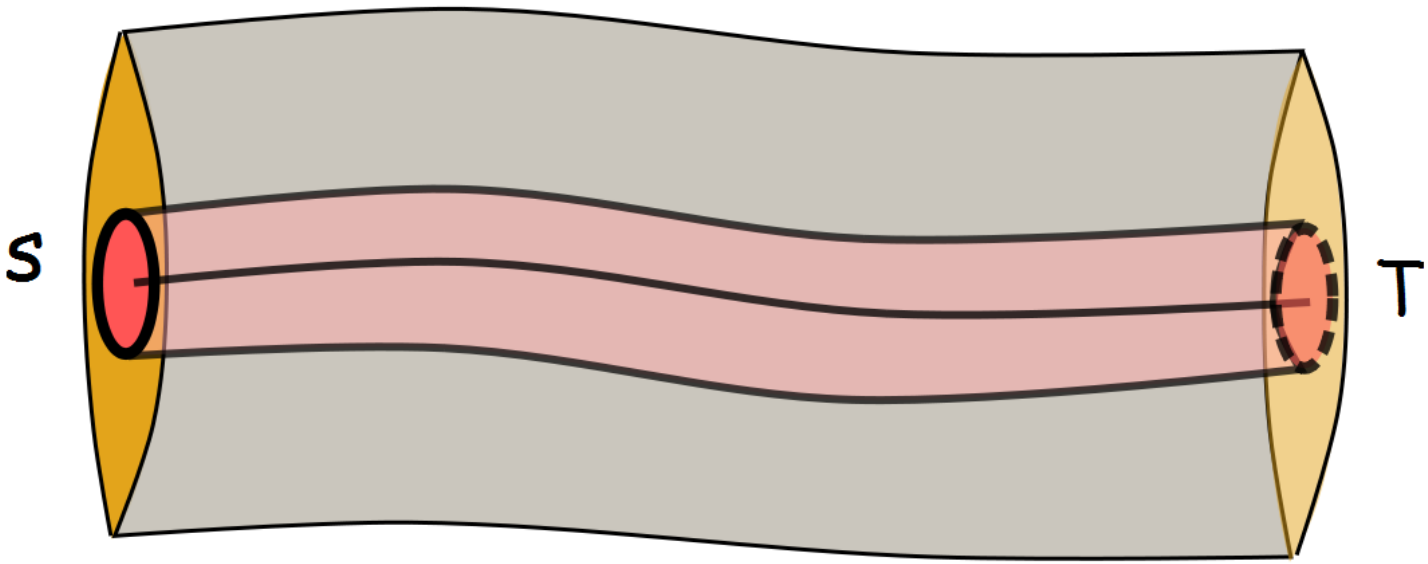
- Aléatoirement?
- Optimiser le cut le plus gros?

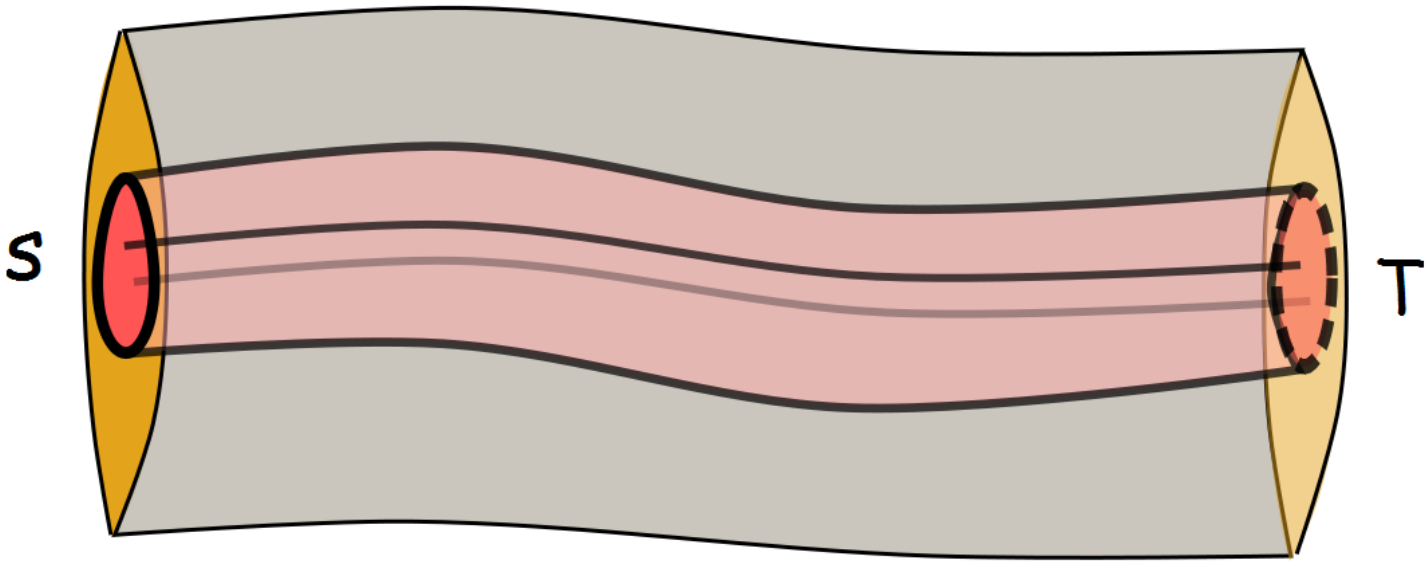
Pas assez de temps pour faire des comparaisons expérimentales

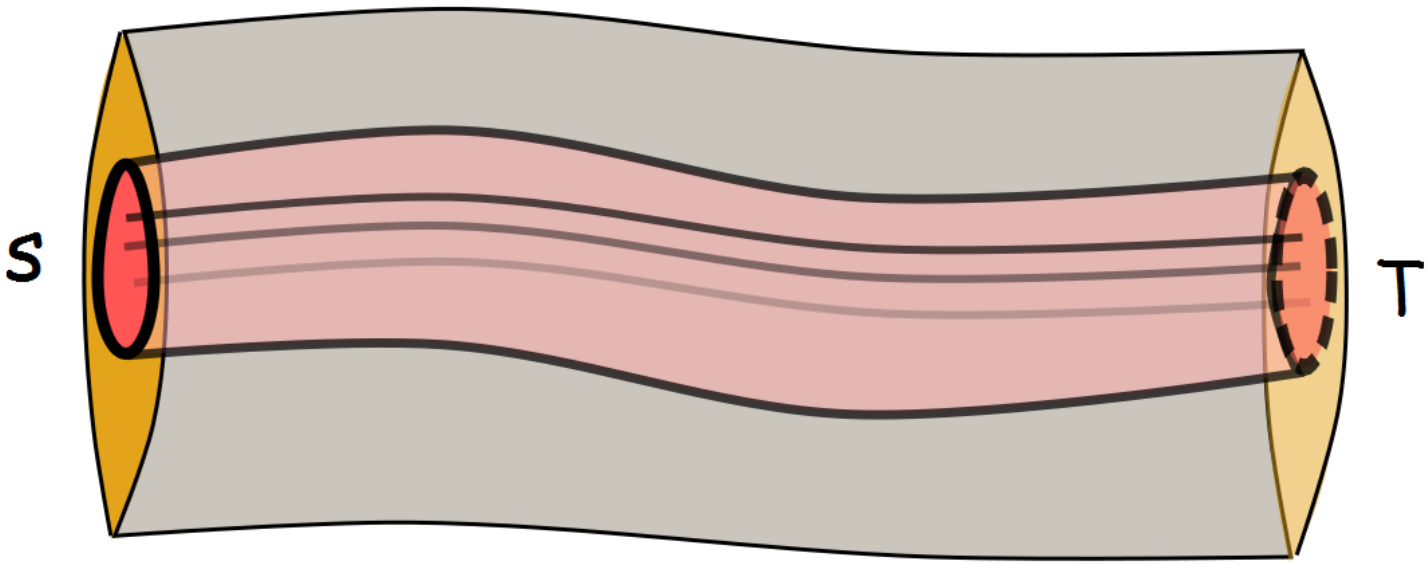
Algorithme de Ford Fulkerson avec recherche par BFS en $O(V\ E^2)$ $= O(V^3)$
Ici on cherche un flot maximum dans un volume sans "trou", peut-on faire mieux?

Si on a déjà trouvé un chemin p de s à t, on peut restreindre la recherche des chemins suivants à un "voisinage" de p, et donc visiter moins de sommets


S
T


S
T


S
T


S
T

Algorithme des voisinages

Si p est un chemin le voisinage de p est noté $\mathcal{N}_p$
L'ensemble N est le voisinage "global", union des voisinages des chemins trouvés

Algorithme des voisinages

Soit p_i un plus court chemin de s à t ;
$N \leftarrow \mathcal{N}_{p_i}$;
$f \leftarrow 0$;
tant que *Il y a un chemin augmentant p de s à t dans N* **faire**
    Augmenter le flot f selon p ;
    $N \leftarrow N \cup \mathcal{N}_p$;
fin

Complexité

Complexité (avec voxels)

L'algorithme des voisinages pour optimiser un cut d'aire C a comme complexité $O(\ell\ C^2)$, où ℓ est la longueur d'un plus court chemin de s à t

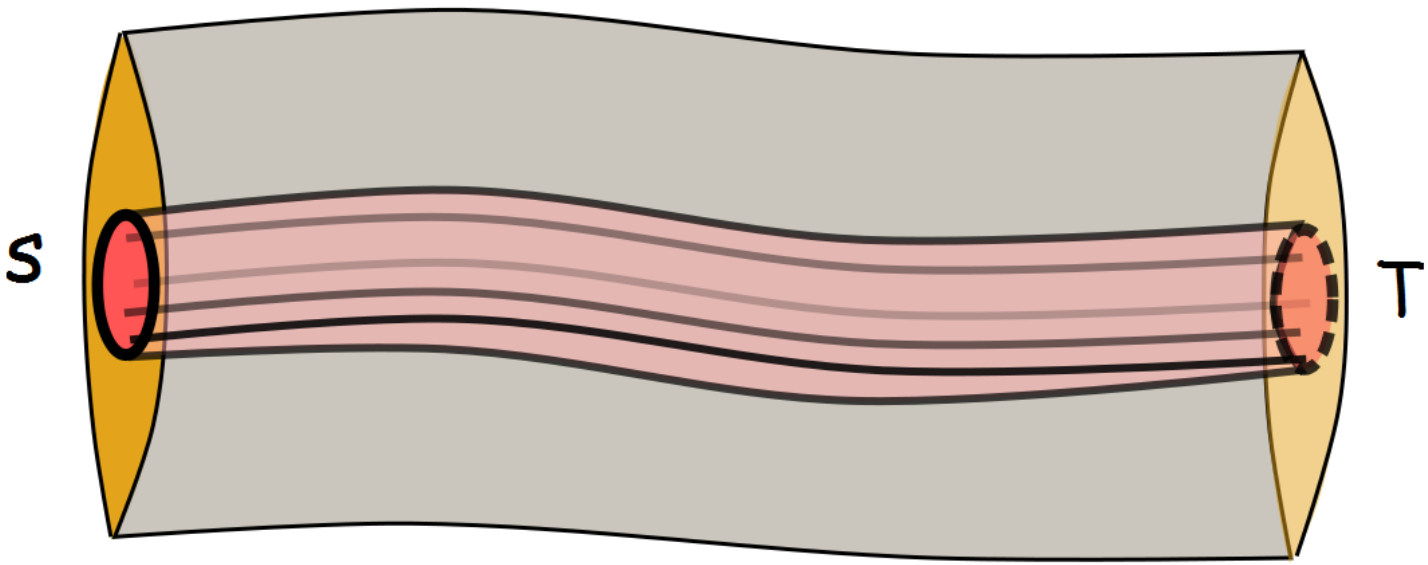
Preuve

Tout au long de l'algorithme, la taille du voisinage est $O(\ell\ C)$ et il y a C BFS

Voisinages par étapes

On peut aussi augmenter le voisinage par "étapes", uniquement quand on ne trouve plus de chemin dans le voisinage en cours

Voisinages par étapes



Vue d'ensemble du code

- Environ 4000 lignes C++ (moi et Jean Marie Favreau)

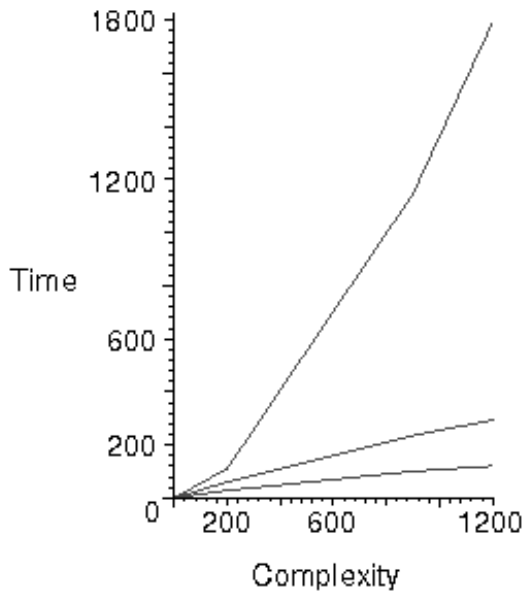
Vue d'ensemble du code

- Environ 4000 lignes C++ (moi et Jean Marie Favreau)
- Un script Python pour Blender (automatisation)

Vue d'ensemble du code

- Environ 4000 lignes C++ (moi et Jean Marie Favreau)
- Un script Python pour Blender (automatisation)
- Tests sur des exemples...

Complexité	Ford Fulkerson	NA	SNA
200K	111s	61s	29s
900K	1147s	236s	101s
1750K	2975s	318s	158s



Ce dont je n'ai pas parlé

- Découpage de surface

Ce dont je n'ai pas parlé

- Découpage de surface
- NP difficulté (Éric Colin de Verdière)

Ce dont je n'ai pas parlé

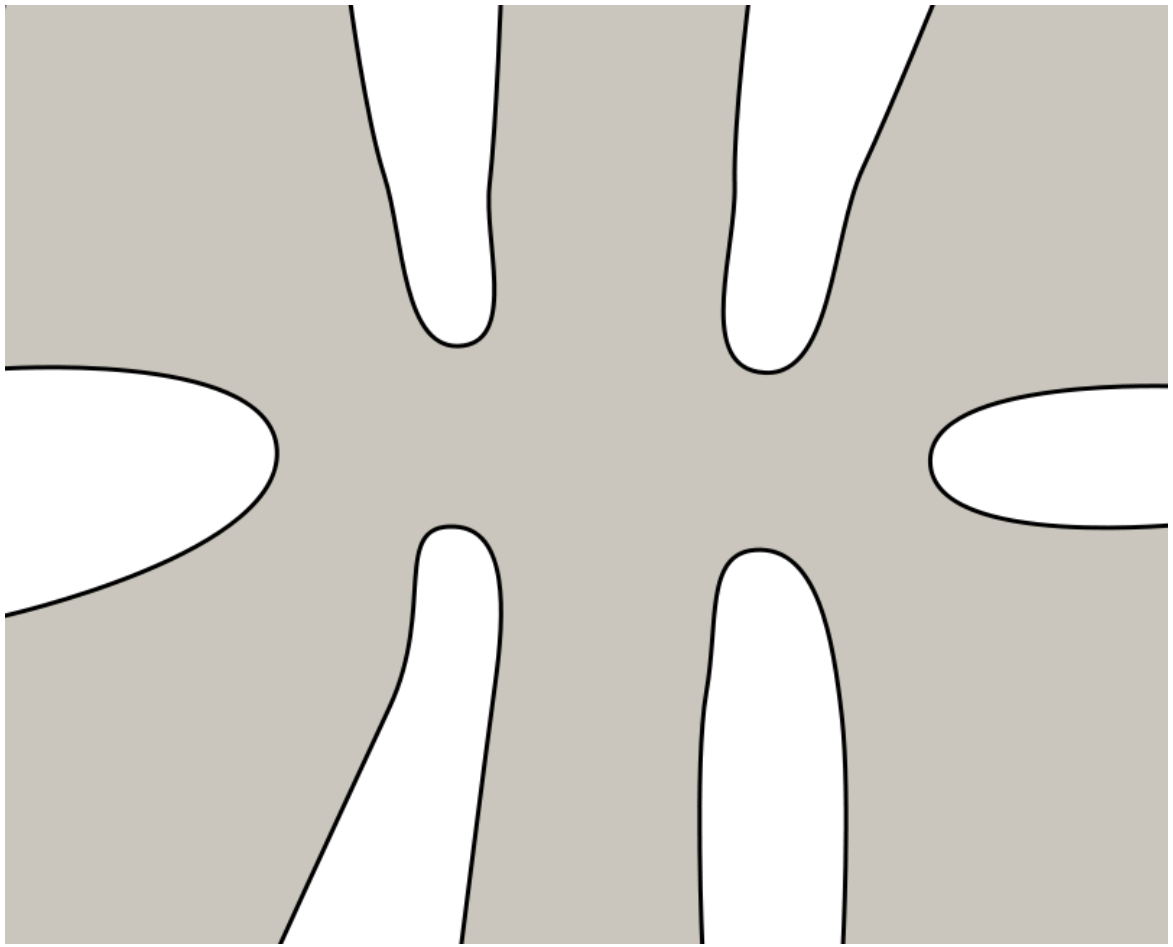
- Découpage de surface
- NP difficulté (Éric Colin de Verdière)
- Détail de l'implémentation

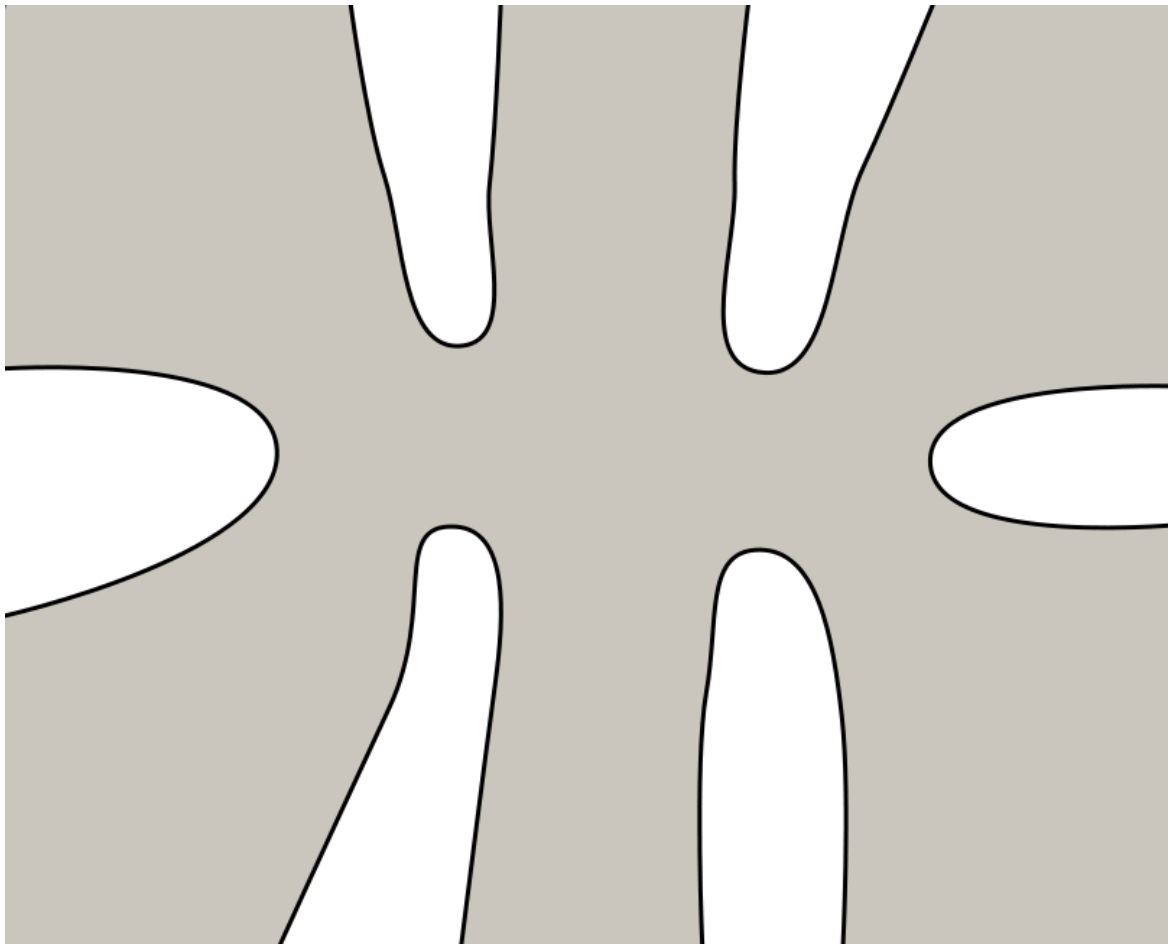
Ce dont je n'ai pas parlé

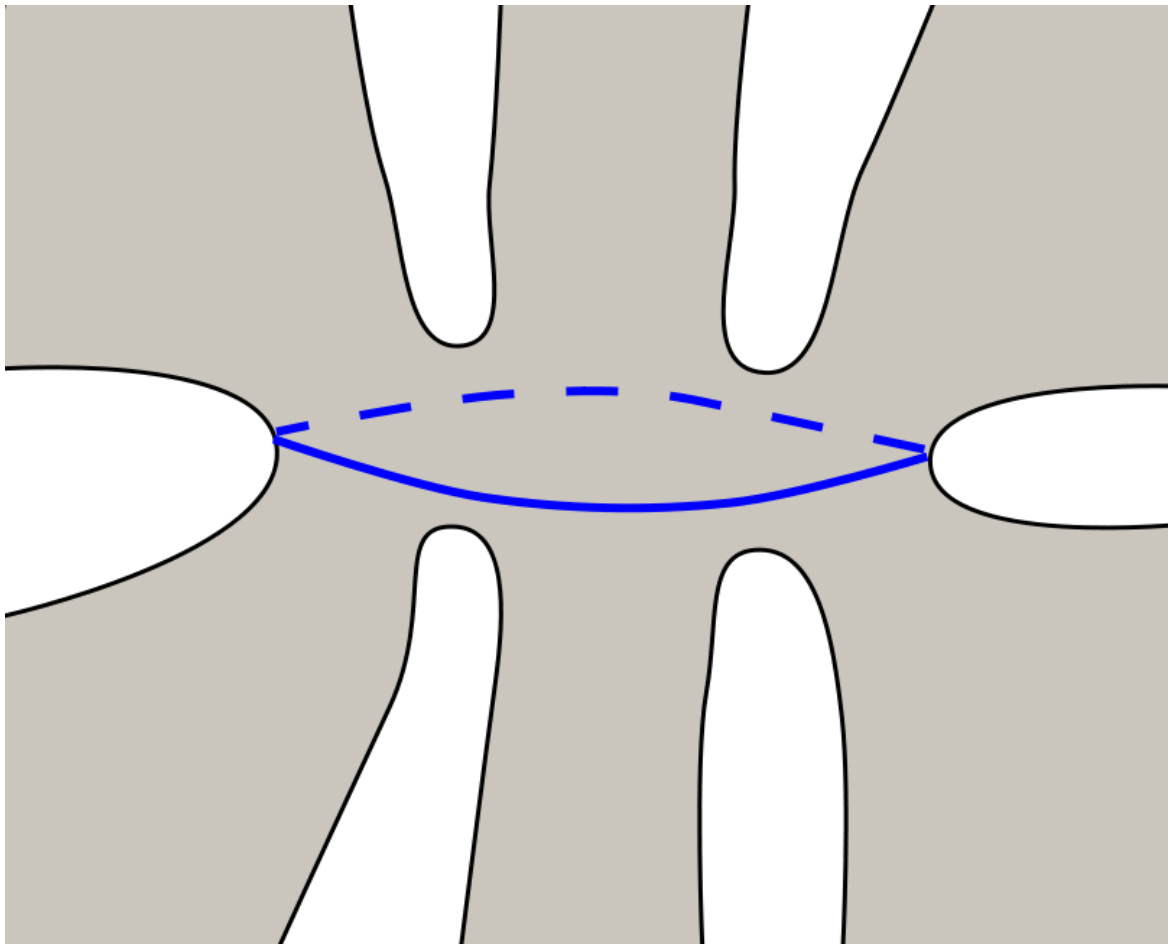
- Découpage de surface
- NP difficulté (Éric Colin de Verdière)
- Détail de l'implémentation
- Applications (RECODS, projet australien)

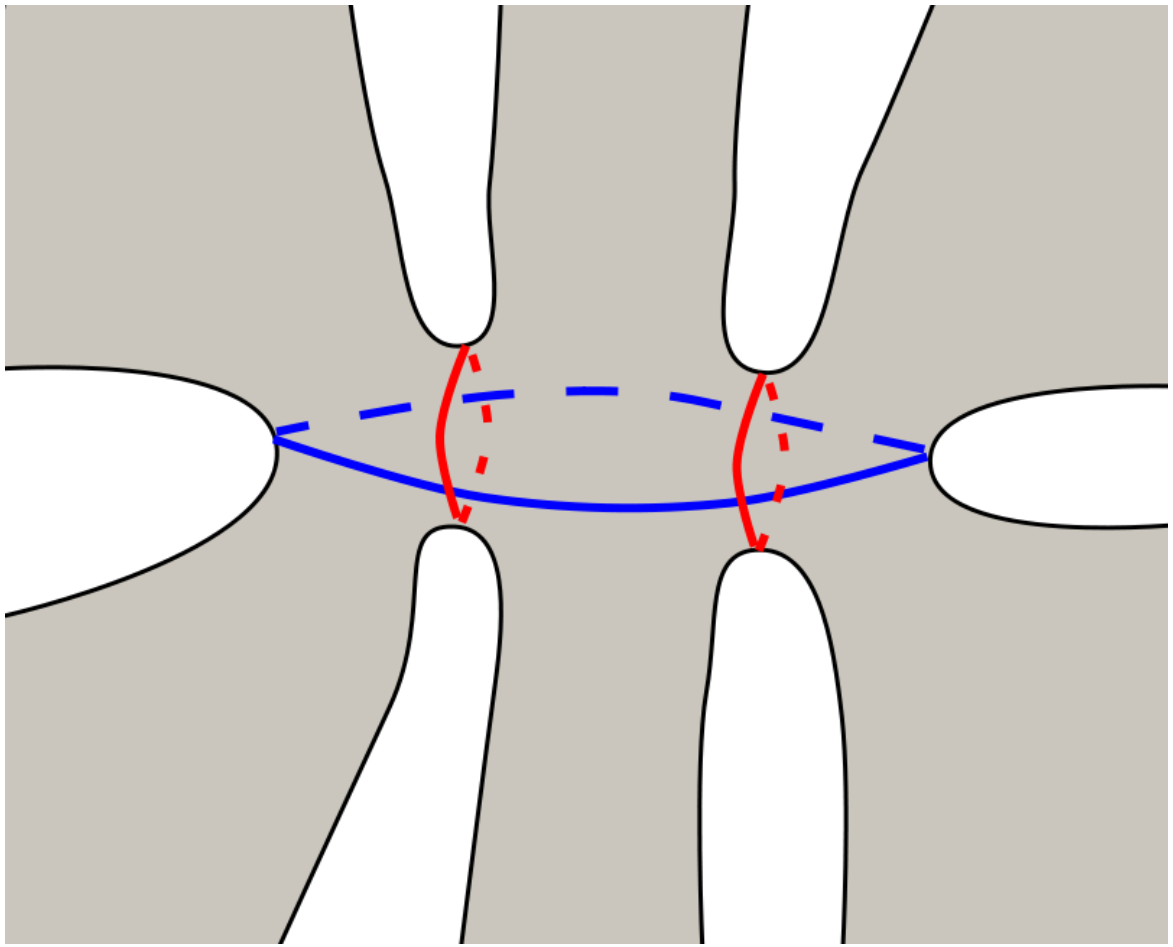
Merci de votre attention.
Avez vous des questions?

L'ordre des optimisations est important, à cause des intersections









Comment orienter un cut?