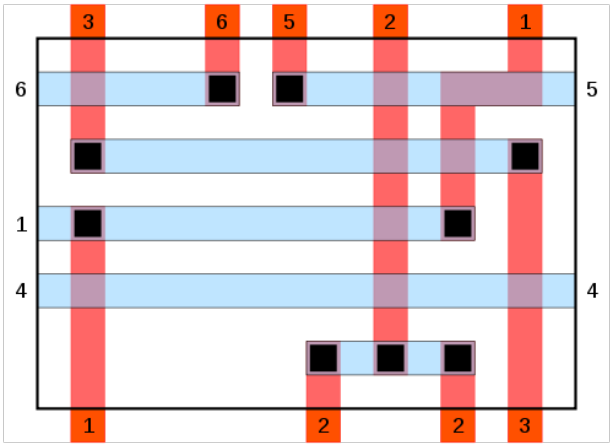
PROJET n°6 *Routage de canaux dans les circuits intégrés*

Un circuit intégré comprend de nombreux éléments fonctionnels qui doivent être interconnectés. Ces interconnexions doivent, bien entendu, être réalisées sans créer de court-circuits. Il faut donc les plannifier en conséquence. Une abstraction qui, dans ce domaine, a eu beaucoup de succès est le *routage de canaux*. Cette abstraction a depuis été déplacée par d'autres, mais elle reste une excellente introduction à la problématique et offre des défis très intéressants pour la programmation par contraintes.

1 Présentation du problème

Dans sa forme générale, un "routeur de canaux" est une sorte d'*échangeur* (comme sur une autoroute) de forme rectangulaire. Sur les bords *nord* et *sud* sont disposés un certain nombre de *plots*. Sur son bord *ouest* se trouve une *ligne* d'entrée, et sur son bord *est* une ligne de sortie.

Un problème de routage de canaux est spécifié par la donnée d'un ensemble de *sous-réseaux*. Un sous-réseau est un ensemble de plots (nord et sud) qui doivent être connectés entre eux. On doit aussi donner l'ensemble des plots qui doivents être connectés à la ligne d'entrée (resp. à la ligne de sortie). Voici un exemple, pris sur wikipedia :



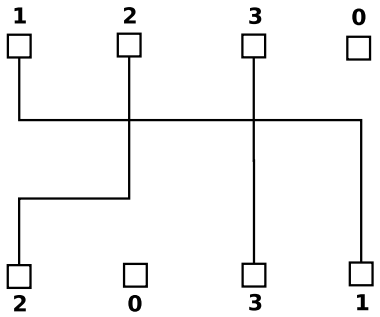
Cet exemple illustre bien la technique utilisée pour créer des interconnexions sans court-circuits : les liaisons sont disposées sur une grille, mais les liaisons horizontales et verticales sont disposées sur des *niveaux* distincts de sorte qu'elles peuvent se croiser sans se court-circuiter. Par exemple, on peut imaginer qu'on utilise les 2 faces d'un circuit intégré ; la face inférieure pour les liaisons verticales, et la face supérieure pour les liaisons horizontales. L'image ci-dessus offre

une solution plus générale, car elle utilise des liaisons horizontales sur les 2 faces. D'une manière générale, on pourrait aussi imaginer avoir plus de 2 niveaux.

Pour notre projet, nous allons faire certaines simplifications. (1) nous allons ignorer les lignes d'entrée et de sortie ; nous nous intéresserons uniquement à connecter les sous-réseaux. (2) nous utiliserons uniquement 2 niveaux, avec les liaison verticales sur le premier et les liaisons horizontales sur le second. (3) pour chaque sous-réseau, nous considérerons des formes d'interconnexion limitées :

- avec une seule branche horizontale. C'est le type de configuration qu'on appelle "no dog-legs."
- avec une ou deux branches horizontales. C'est la configuration optionellement "dog-leg".

Voici un exemple de circuit :



chaque plot est annoté avec un numéro qui représente le sous-réseau auquel il est connecté. Le numéro 0 indique que le plot n'est pas utilisé. Dans le fichier de données, un tel problème sera specifié de la manière suivante :

```
Dims = n x k
Haut = N1 N2 ... Nk
Bas = M1 M2 ... Mk
```

ou `n` est le nombre de rangées horizontales (en comptant les 2 rangées de plots), `k` est le nombre de colonnes verticales (c'est à dire le nombre de plots du haut (ou du bas puisqu'il y en a autant)). `Haut` (resp. `Bas`) donne les numéros de sous-réseaux pour le plots du haut (resp. du bas). Pour l'exemple ci-dessus, on aurait :

```
Dims = 4 x 4
Haut = 1 2 3 0
Bas  = 2 0 3 1
```

Le fichier `projet06.txt` contient quelques instances de problèmes. Le script python `projet06.py` illustre comment charger ces instances, et les affiche dans le format YAML. A vous de modifier ce script pour créer des fichier de données appropriés pour Minizinc.