

# Projet Info LDD2 – TD 4

Renaud Vilmar -- vilmar@lsv.fr

**Objectifs du TP :** Sauvegarde et représentation de graphes.

Dans ce TD, on va se doter de méthodes pour sauvegarder nos graphes, mais également les visualiser. Dans la suite on propose d'utiliser le format `.dot` (ou `.gv`), mais vous êtes libre de dévier de ce choix.

La syntaxe de base du format `.dot` est très simple, voici un exemple ci-dessous

---

```
digraph G {
    v0 [label="&"];
    v1 [label="-"];
    v4 [label="|"];
    v0 -> v1 -> v2;
    v0 -> v3;
    v2 -> v3;
    v2 -> v3;
    v3 -> v4;
    v2 -> v4;
}
```

---

On peut ensuite visualiser ce graphe avec un outil approprié. Par exemple avec la ligne de commande

---

```
dot -Tpdf <input_file.dot> -o <output_file.pdf>
```

---

(on peut remplacer le format de sortie par `png`, `svg`, `ps`, ...) si `graphviz` est installé sur le système. On peut toujours utiliser un visualisateur en ligne, e.g. <https://dreampuf.github.io/GraphvizOnline/>. On peut trouver plus d'infos sur le format `.dot` par exemple ici <https://www.graphviz.org/>.

▮ **Exercice 1 :** Écrire une méthode :

`save_as_dot_file(self, path, verbose=False)`

qui va enregistrer le graphe en question en format `.dot` à l'endroit spécifié par `path`.

Vous avez peut-être remarqué que si on spécifiait un `label` pour un noeud, celui-ci n'affichait plus son `id`. Lorsque `verbose=True`, on veut afficher le `label` et l'`id` (ça simplifie le débogage).

Le format permet donner des attributs arbitraires aux noeuds, qui seront simplement ignorés si non-reconnus. On peut s'en servir pour donner l'information qu'un noeud est une entrée ou une sortie.

Si vous avez décidé d'utiliser un autre format pour enregistrer les graphes, adaptez cette méthode. ▮

▮ **Exercice 2 :** Faire le chemin inverse : implémenter une méthode de classe `from_dot_file` qui lit un fichier `.dot` et crée un `open_digraph` à partir de lui

(enregistrer un graphe avec `verbose=False` puis le charger devrait donner le même graphe qu'au départ). ┘

┐ **Exercice 3 :** Créer une méthode `display` (avec la variable optionnelle `verbose=False`) qui affiche directement le graphe. On pourra utiliser `os.system(<ligne de commande>)` pour lancer la ligne de commande depuis python.

- Si vous avez moyen de visualiser le graphe via ligne de commande, il suffit de créer le fichier `.dot` à un endroit temporaire, créer un pdf/png/... et l'ouvrir
- Si vous souhaitez utiliser un visualiseur en ligne, on peut utiliser en ligne de commande par exemple :

---

```
firefox -url https://dreampuf.github.io/GraphvizOnline/#digraph{v0 --> v1}
```

---

(notez que les sauts de ligne ont disparu (ou sont remplacés par `%0A%09`) et que les `;` sont remplacés par `%3B`). ┘