Impression équilibrée

Informations

Le projet peut être compilé sur Windows (avec Visual studio) et sur Linux (avec la commande *make*), il est réalisé en C++.

1. Sous Windows

Cliquer sur li325.sln pour ouvrir le projet dans Visual studio 2012.

1. Sous Linux

Les sources peuvent être éditées avec n’importe quel éditeur de texte, la compilation s’effectue en utilisant la commande *make* à la racine du projet.

Il est important de comprendre la responsabilité de chaque classe :

1. TList<T>

Il s’agit d’une classe implémentant la structure de liste générique.

1. Range

Cette classe décrit un intervalle, dans notre cas un intervalle de mots.

1. Printer

Cette classe représente l’imprimante.

1. Problem

Il s’agit de la classe qui représente le problème général, elle sert de base pour GloutonProblem et DynamicProblem.

1. GloutonProblem

Une implémentation du problème utilisant un algorithme Glouton.

1. DynamicProblem

Une implémentation du problème utilisant un algorithme Dynamique.

1. String

Une classe facilitant la manipulation des chaines de caractère.

Les fichiers d’en tête de chacune de ces classes contiennent des commentaires donnant plus de détails sur leur fonctionnement.

Le fichier *main.cpp* créé et résout 12 problèmes, 6 problèmes goutons et 6 problèmes dynamiques (avec des paramètres différents). Pour cela le programme attend 2 paramètres : le fichier d’entrée et le fichier de sortie. Le fichier d’entrée doit contenir le texte, le fichier de sortie contiendra les solutions ainsi que quelques informations supplémentaires.

Exemple :

*li325.exe input.txt output.txt*

Questions

1. L’algorithme de Glouton est disponible dans la fonction *Solve* de la classe *GloutonProblem.* Cet algorithme ne fournit pas l’optimum, voici un contre-exemple :

**Algorithme de Glouton appliqué au texte avec W = 20**

LINES BADNESS

----------------------------

Inventée\_par\_le\_\_\_\_\_ 125

professeur\_Richard\_\_ 8

Bellman,\_la\_\_\_\_\_\_\_\_\_ 729

programmation\_\_\_\_\_\_\_ 343

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 27

**d'un\_ordinateur\_tout 0**

**problème\_\_\_\_\_\_\_\_\_\_\_\_ 1728**

....................

Le score total peut être amélioré:

de

....................

**d’un\_ordinateur\_tout 0**

**problème\_\_\_\_\_\_\_\_\_\_\_\_ 1728**

....................

à

....................

**d’un\_ordinateur\_\_\_\_\_ 125**

**tout\_problème\_\_\_\_\_\_\_ 343**

....................

L'algorithme de glouton nous fournit donc une heuristique.

1. L’algorithme Dynamique est disponible dans la fonction *Solve* de la classe *DynamicProblem.*

Notations :

W : capacité d’une ligne.

Words : tableau de mots.

n : taille du tableau *Words.*

Badness : fonction du calcul du score, plus il est bas, meilleur il est.

DP[i] : structure optimale, contient le score *Badness* minimum réalisable en prenant en compte les mots *Words*[i : n].

1. Analyse de la complexité temporelle :

Nous cherchons donc à calculer DP[0], le tableau DP est défini comme tel :

DP[n+1] = 0

DP[i] = min jϵ[i+1,W] (*Badness*(i, j)+DP[j+1])

Le calcul du *min* se fait en O(*W*).

Le remplissage du tableau *DP* se fera en O(*n*) maximum (Si la valeur demandée y est absente, alors un appel récursif est engendré).

Cela nous donne un total de O(Wn).

Remarque : La complexité est en fait inférieure puisque le calcul du *min* est restreint au nombre maximum de mots entrants sur une ligne.

1. Analyse de la complexité en mémoire :

La structure optimale ne demande qu’un tableau de *n* entiers.

Au plus n appels récursifs seront exécutés.

Au total (n\*4 + taille de la fonction *DP*) octets seront nécessaires.

1. Ne prendre que la somme des caractères améliorera le temps nécessaire au calcul du score *Badness*. Cela ne change rien pour l’algorithme résolvant le problème.
2. L’algorithme n’a pas eu besoin de modification, seuls les paramètres du problème ont changés (voir annexe).

Annexe

Voici les différents résultats retournés pour le texte et les paramètres demandés :

Glouton Problem W=20 Power=1

LINES BADNESS

----------------------------

Inventée\_par\_le\_\_\_\_\_ 5

professeur\_Richard\_\_ 2

Bellman,\_la\_\_\_\_\_\_\_\_\_ 9

programmation\_\_\_\_\_\_\_ 7

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 3

d’un\_ordinateur\_tout 0

problème\_\_\_\_\_\_\_\_\_\_\_\_ 12

d’optimisation\_dont\_ 1

la\_fonction\_objectif 0

se\_décrit\_comme\_la\_\_ 2

somme\_de\_fonctions\_\_ 2

monotones\_\_\_\_\_\_\_\_\_\_\_ 11

non-décroissantes\_\_\_ 3

des\_ressources.\_\_\_\_\_ 5

Concrètement,\_cela\_\_ 2

signifie\_que\_l’on\_va 0

pouvoir\_déduire\_la\_\_ 2

solution\_optimale\_\_\_ 3

d’un\_problème\_à\_\_\_\_\_ 5

partir\_d’une\_\_\_\_\_\_\_\_ 8

solution\_optimale\_\_\_ 3

d’un\_sous\_problème.\_ 1

On\_appelle\_\_\_\_\_\_\_\_\_\_ 10

algorithme\_glouton\_\_ 2

un\_algorithme\_qui\_\_\_ 3

suit\_le\_principe\_de\_ 1

faire,\_étape\_par\_\_\_\_ 4

étape,\_un\_choix\_\_\_\_\_ 5

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 2

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_\_ 2

fournit\_pas\_\_\_\_\_\_\_\_\_ 9

systématiquement\_la\_ 1

solution\_optimale,\_\_ 2

il\_est\_appelé\_une\_\_\_ 3

heuristique\_\_\_\_\_\_\_\_\_ 9

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 142

Glouton Problem W=20 Power=2

LINES BADNESS

----------------------------

Inventée\_par\_le\_\_\_\_\_ 25

professeur\_Richard\_\_ 4

Bellman,\_la\_\_\_\_\_\_\_\_\_ 81

programmation\_\_\_\_\_\_\_ 49

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 9

d’un\_ordinateur\_tout 0

problème\_\_\_\_\_\_\_\_\_\_\_\_ 144

d’optimisation\_dont\_ 1

la\_fonction\_objectif 0

se\_décrit\_comme\_la\_\_ 4

somme\_de\_fonctions\_\_ 4

monotones\_\_\_\_\_\_\_\_\_\_\_ 121

non-décroissantes\_\_\_ 9

des\_ressources.\_\_\_\_\_ 25

Concrètement,\_cela\_\_ 4

signifie\_que\_l’on\_va 0

pouvoir\_déduire\_la\_\_ 4

solution\_optimale\_\_\_ 9

d’un\_problème\_à\_\_\_\_\_ 25

partir\_d’une\_\_\_\_\_\_\_\_ 64

solution\_optimale\_\_\_ 9

d’un\_sous\_problème.\_ 1

On\_appelle\_\_\_\_\_\_\_\_\_\_ 100

algorithme\_glouton\_\_ 4

un\_algorithme\_qui\_\_\_ 9

suit\_le\_principe\_de\_ 1

faire,\_étape\_par\_\_\_\_ 16

étape,\_un\_choix\_\_\_\_\_ 25

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 4

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_\_ 4

fournit\_pas\_\_\_\_\_\_\_\_\_ 81

systématiquement\_la\_ 1

solution\_optimale,\_\_ 4

il\_est\_appelé\_une\_\_\_ 9

heuristique\_\_\_\_\_\_\_\_\_ 81

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 934

Glouton Problem W=20 Power=3

LINES BADNESS

----------------------------

Inventée\_par\_le\_\_\_\_\_ 125

professeur\_Richard\_\_ 8

Bellman,\_la\_\_\_\_\_\_\_\_\_ 729

programmation\_\_\_\_\_\_\_ 343

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 27

d’un\_ordinateur\_tout 0

problème\_\_\_\_\_\_\_\_\_\_\_\_ 1728

d’optimisation\_dont\_ 1

la\_fonction\_objectif 0

se\_décrit\_comme\_la\_\_ 8

somme\_de\_fonctions\_\_ 8

monotones\_\_\_\_\_\_\_\_\_\_\_ 1331

non-décroissantes\_\_\_ 27

des\_ressources.\_\_\_\_\_ 125

Concrètement,\_cela\_\_ 8

signifie\_que\_l’on\_va 0

pouvoir\_déduire\_la\_\_ 8

solution\_optimale\_\_\_ 27

d’un\_problème\_à\_\_\_\_\_ 125

partir\_d’une\_\_\_\_\_\_\_\_ 512

solution\_optimale\_\_\_ 27

d’un\_sous\_problème.\_ 1

On\_appelle\_\_\_\_\_\_\_\_\_\_ 1000

algorithme\_glouton\_\_ 8

un\_algorithme\_qui\_\_\_ 27

suit\_le\_principe\_de\_ 1

faire,\_étape\_par\_\_\_\_ 64

étape,\_un\_choix\_\_\_\_\_ 125

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 8

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_\_ 8

fournit\_pas\_\_\_\_\_\_\_\_\_ 729

systématiquement\_la\_ 1

solution\_optimale,\_\_ 8

il\_est\_appelé\_une\_\_\_ 27

heuristique\_\_\_\_\_\_\_\_\_ 729

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 7906

Dynamic Problem W=20 Power=1

LINES BADNESS

----------------------------

Inventée\_\_\_\_\_\_\_\_\_\_\_\_ 12

par\_le\_professeur\_\_\_ 3

Richard\_Bellman,\_\_\_\_ 4

la\_programmation\_\_\_\_ 4

dynamique\_permet\_\_\_\_ 4

de\_résoudre\_au\_moyen 0

d’un\_ordinateur\_\_\_\_\_ 5

tout\_problème\_\_\_\_\_\_\_ 7

d’optimisation\_\_\_\_\_\_ 6

dont\_la\_fonction\_\_\_\_ 4

objectif\_se\_décrit\_\_ 2

comme\_la\_somme\_de\_\_\_ 3

fonctions\_monotones\_ 1

non-décroissantes\_\_\_ 3

des\_ressources.\_\_\_\_\_ 5

Concrètement,\_\_\_\_\_\_\_ 7

cela\_signifie\_\_\_\_\_\_\_ 7

que\_l’on\_va\_pouvoir\_ 1

déduire\_la\_solution\_ 1

optimale\_d’un\_\_\_\_\_\_\_ 7

problème\_à\_partir\_\_\_ 3

d’une\_solution\_\_\_\_\_\_ 6

optimale\_d’un\_sous\_\_ 2

problème.\_On\_appelle 0

algorithme\_glouton\_\_ 2

un\_algorithme\_\_\_\_\_\_\_ 7

qui\_suit\_le\_principe 0

de\_faire,\_étape\_\_\_\_\_ 5

par\_étape,\_un\_choix\_ 1

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 2

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_\_\_\_ 4

cas\_où\_l’algorithme\_ 1

ne\_fournit\_pas\_\_\_\_\_\_ 6

systématiquement\_la\_ 1

solution\_optimale,\_\_ 2

il\_est\_appelé\_\_\_\_\_\_\_ 7

une\_heuristique\_\_\_\_\_ 5

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 142

Dynamic Problem W=20 Power=2

LINES BADNESS

----------------------------

Inventée\_par\_\_\_\_\_\_\_\_ 64

le\_professeur\_\_\_\_\_\_\_ 49

Richard\_Bellman,\_\_\_\_ 16

la\_programmation\_\_\_\_ 16

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 9

d’un\_ordinateur\_\_\_\_\_ 25

tout\_problème\_\_\_\_\_\_\_ 49

d’optimisation\_\_\_\_\_\_ 36

dont\_la\_fonction\_\_\_\_ 16

objectif\_se\_décrit\_\_ 4

comme\_la\_somme\_de\_\_\_ 9

fonctions\_monotones\_ 1

non-décroissantes\_\_\_ 9

des\_ressources.\_\_\_\_\_ 25

Concrètement,\_\_\_\_\_\_\_ 49

cela\_signifie\_que\_\_\_ 9

l’on\_va\_pouvoir\_\_\_\_\_ 25

déduire\_la\_solution\_ 1

optimale\_d’un\_\_\_\_\_\_\_ 49

problème\_à\_partir\_\_\_ 9

d’une\_solution\_\_\_\_\_\_ 36

optimale\_d’un\_sous\_\_ 4

problème.\_On\_appelle 0

algorithme\_glouton\_\_ 4

un\_algorithme\_qui\_\_\_ 9

suit\_le\_principe\_\_\_\_ 16

de\_faire,\_étape\_\_\_\_\_ 25

par\_étape,\_un\_choix\_ 1

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 4

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_\_\_\_ 16

cas\_où\_l’algorithme\_ 1

ne\_fournit\_pas\_\_\_\_\_\_ 36

systématiquement\_la\_ 1

solution\_optimale,\_\_ 4

il\_est\_appelé\_\_\_\_\_\_\_ 49

une\_heuristique\_\_\_\_\_ 25

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 704

Dynamic Problem W=20 Power=3

LINES BADNESS

----------------------------

Inventée\_par\_\_\_\_\_\_\_\_ 512

le\_professeur\_\_\_\_\_\_\_ 343

Richard\_Bellman,\_\_\_\_ 64

la\_programmation\_\_\_\_ 64

dynamique\_permet\_de\_ 1

résoudre\_au\_moyen\_\_\_ 27

d’un\_ordinateur\_\_\_\_\_ 125

tout\_problème\_\_\_\_\_\_\_ 343

d’optimisation\_\_\_\_\_\_ 216

dont\_la\_fonction\_\_\_\_ 64

objectif\_se\_décrit\_\_ 8

comme\_la\_somme\_de\_\_\_ 27

fonctions\_monotones\_ 1

non-décroissantes\_\_\_ 27

des\_ressources.\_\_\_\_\_ 125

Concrètement,\_\_\_\_\_\_\_ 343

cela\_signifie\_que\_\_\_ 27

l’on\_va\_pouvoir\_\_\_\_\_ 125

déduire\_la\_solution\_ 1

optimale\_d’un\_\_\_\_\_\_\_ 343

problème\_à\_partir\_\_\_ 27

d’une\_solution\_\_\_\_\_\_ 216

optimale\_d’un\_sous\_\_ 8

problème.\_On\_appelle 0

algorithme\_glouton\_\_ 8

un\_algorithme\_qui\_\_\_ 27

suit\_le\_principe\_\_\_\_ 64

de\_faire,\_étape\_\_\_\_\_ 125

par\_étape,\_un\_choix\_ 1

optimum\_local,\_dans\_ 1

l’espoir\_d’obtenir\_\_ 8

un\_résultat\_optimum\_ 1

global.\_Dans\_les\_\_\_\_ 64

cas\_où\_l’algorithme\_ 1

ne\_fournit\_pas\_\_\_\_\_\_ 216

systématiquement\_la\_ 1

solution\_optimale,\_\_ 8

il\_est\_appelé\_\_\_\_\_\_\_ 343

une\_heuristique\_\_\_\_\_ 125

gloutonne.\_\_\_\_\_\_\_\_\_\_ 0

----------------------------

TOTAL 4030

Glouton Problem W=50 Power=1

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_la\_\_\_\_ 4

programmation\_dynamique\_permet\_de\_résoudre\_au\_\_\_\_\_ 5

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_la\_somme 0

de\_fonctions\_monotones\_non-décroissantes\_des\_\_\_\_\_\_ 6

ressources.\_Concrètement,\_cela\_signifie\_que\_l’on\_\_ 2

va\_pouvoir\_déduire\_la\_solution\_optimale\_d’un\_\_\_\_\_\_ 6

problème\_à\_partir\_d’une\_solution\_optimale\_d’un\_\_\_\_ 4

sous\_problème.\_On\_appelle\_algorithme\_glouton\_un\_\_\_ 3

algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape\_\_\_ 3

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 2

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 2

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 37

Glouton Problem W=50 Power=2

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_la\_\_\_\_ 16

programmation\_dynamique\_permet\_de\_résoudre\_au\_\_\_\_\_ 25

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_la\_somme 0

de\_fonctions\_monotones\_non-décroissantes\_des\_\_\_\_\_\_ 36

ressources.\_Concrètement,\_cela\_signifie\_que\_l’on\_\_ 4

va\_pouvoir\_déduire\_la\_solution\_optimale\_d’un\_\_\_\_\_\_ 36

problème\_à\_partir\_d’une\_solution\_optimale\_d’un\_\_\_\_ 16

sous\_problème.\_On\_appelle\_algorithme\_glouton\_un\_\_\_ 9

algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape\_\_\_ 9

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 4

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 4

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 159

Glouton Problem W=50 Power=3

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_la\_\_\_\_ 64

programmation\_dynamique\_permet\_de\_résoudre\_au\_\_\_\_\_ 125

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_la\_somme 0

de\_fonctions\_monotones\_non-décroissantes\_des\_\_\_\_\_\_ 216

ressources.\_Concrètement,\_cela\_signifie\_que\_l’on\_\_ 8

va\_pouvoir\_déduire\_la\_solution\_optimale\_d’un\_\_\_\_\_\_ 216

problème\_à\_partir\_d’une\_solution\_optimale\_d’un\_\_\_\_ 64

sous\_problème.\_On\_appelle\_algorithme\_glouton\_un\_\_\_ 27

algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape\_\_\_ 27

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 8

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 8

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 763

Dynamic Problem W=50 Power=1

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_\_\_\_\_\_\_ 7

la\_programmation\_dynamique\_permet\_de\_résoudre\_au\_\_ 2

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_\_\_\_\_\_\_\_\_ 9

la\_somme\_de\_fonctions\_monotones\_non-décroissantes\_ 1

des\_ressources.\_Concrètement,\_cela\_signifie\_\_\_\_\_\_\_ 7

que\_l’on\_va\_pouvoir\_déduire\_la\_solution\_optimale\_\_ 2

d’un\_problème\_à\_partir\_d’une\_solution\_optimale\_\_\_\_ 4

d’un\_sous\_problème.\_On\_appelle\_algorithme\_glouton\_ 1

un\_algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape 0

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 2

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 2

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 37

Dynamic Problem W=50 Power=2

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_la\_\_\_\_ 16

programmation\_dynamique\_permet\_de\_résoudre\_au\_\_\_\_\_ 25

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_la\_\_\_\_\_\_ 36

somme\_de\_fonctions\_monotones\_non-décroissantes\_\_\_\_ 16

des\_ressources.\_Concrètement,\_cela\_signifie\_que\_\_\_ 9

l’on\_va\_pouvoir\_déduire\_la\_solution\_optimale\_d’un\_ 1

problème\_à\_partir\_d’une\_solution\_optimale\_d’un\_\_\_\_ 16

sous\_problème.\_On\_appelle\_algorithme\_glouton\_un\_\_\_ 9

algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape\_\_\_ 9

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 4

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 4

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 145

Dynamic Problem W=50 Power=3

LINES BADNESS

----------------------------------------------------------

Inventée\_par\_le\_professeur\_Richard\_Bellman,\_la\_\_\_\_ 64

programmation\_dynamique\_permet\_de\_résoudre\_au\_\_\_\_\_ 125

moyen\_d’un\_ordinateur\_tout\_problème\_d’optimisation 0

dont\_la\_fonction\_objectif\_se\_décrit\_comme\_la\_\_\_\_\_\_ 216

somme\_de\_fonctions\_monotones\_non-décroissantes\_\_\_\_ 64

des\_ressources.\_Concrètement,\_cela\_signifie\_que\_\_\_ 27

l’on\_va\_pouvoir\_déduire\_la\_solution\_optimale\_d’un\_ 1

problème\_à\_partir\_d’une\_solution\_optimale\_d’un\_\_\_\_ 64

sous\_problème.\_On\_appelle\_algorithme\_glouton\_un\_\_\_ 27

algorithme\_qui\_suit\_le\_principe\_de\_faire,\_étape\_\_\_ 27

par\_étape,\_un\_choix\_optimum\_local,\_dans\_l’espoir\_\_ 8

d’obtenir\_un\_résultat\_optimum\_global.\_Dans\_les\_cas 0

où\_l’algorithme\_ne\_fournit\_pas\_systématiquement\_la 0

solution\_optimale,\_il\_est\_appelé\_une\_heuristique\_\_ 8

gloutonne.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 0

----------------------------------------------------------

TOTAL 631

On remarque que l’algorithme Glouton est insensible à la valeur du paramètre *power* (évidant).

L’algorithme dynamique, qui nous retourne l’optimum est sensible au paramètre *power*, plus celui-ci est élevé moins violents sont les espacements (ils sont plus réguliers) ce qui est intéressant si l’on souhaite « étendre » chaque ligne en espaçant plus ou moins les mots.

Je me suis inspiré du cours suivant : http://www.youtube.com/watch?v=ENyox7kNKeY.