Modèles linéaires - DM

Pather Stevenson, Larafi Zakaria

Enseignant : M. Abdelkafi Omar

L3 Informatique Groupe 1 - Semestre V - 2021

Les transistors d'Ornicar

0 - Préambule

Travail demandé

Il nous est demandé dans ce devoir de proposer une démarche et d'en implémenter et expliquer le modèle pour répondre aux attentes commerciales des transistors d'Ornicar.

Problèmes exposés

On retrouve dans les attentes d'Ornicar. Des problèmes d'affectation classique en recherche opérationnelle et optimisation combinatoire. Il est principalement question ici de former des sous-ensembles de transistors. Qui respectent différents critères en terme de dispersion, notamment en cherchant comment appairer aux mieux un nombre de transistors parmi un ensemble de transistors pour permettre une dispersion acceptable pour une mise sur le marché.

Solution proposée

La solution la plus adaptée que l'on propose. Pour répondre à ce genre de problématique. Tourne autour de l'utilisation de variables, tableaux binaires. Qui permettent de venir généraliser et proposer à l'aide de formules qui contiennent ses valeurs binaires. De venir exprimer des contraintes simples à interprêter mais robuste pour permettre de trouver des solutions optimales à nos problèmes d'affectations.

Outils et langages

Nous avons avant tout chercher résoudre les différentes problématiques rencontrées au travers de la logique combinatoire apprise durant notre cursus. En utilisant des notations et expressions mathématiques pour permettre une compréhension de nos modèles et solutions au plus grand nombres. Et ce détâcher au plus du support du langage informatique que l'on utilise. Ce langage est AMPL. Nous l'utilions donc pour implémenter nos modèles. Et

nous avons utilisé AMPL avec le solveur Gurobi. Qui permet notamment dans son panel de résolution, la résolution de problèmes linéaires composés de données entières.

1 - Présentation du problème

Ornicar est fabricant de composants électroniques. Il dispose en particulier d'un jeu de transistors, qu'il souhaite vendre. Il hésite sur la façon de vendre ses transistors, et vous allez l'aider.

Un transistor est un composant électronique à 3 pattes, qui sert essentiellement à amplifier un courant. Chaque transistor fabriqué par Ornicar est soit de type PNP, soit de type NPN. De plus, chaque transistor a deux caractéristiques principales, nommées hfe (le gain = coefficient d'amplification) et vbe (tension base émetteur). Ces deux caractéristiques sont des valeurs réelles positives non nulles. Pour tout transistor, hfe est entre 0 et $MAXHFE$ (où $MAXHFE$ vaut 600), et vbe est entre 0 et $MAXVBE$ (où $MAXVBE$ vaut 1 Volt).

Il est en général plus avantageux de vendre ensemble des transistors qui disposent de caractéristiques (vbe et hfe) les plus proches possibles. On définit ainsi la dispersion d'un ensemble de transistors $T_1, ..., T_n$, où T_i a un hfe égal à h_i, et un vbe égal à v_i, par la formule suivante :

$$D(T_1, ..., T_n) = \max\left(\frac{d_h}{MAXHFE}, \frac{d_v}{MAXVBE}\right)$$

où $d_h = \max_i(h_i) - \min_i(h_i)$ et $d_v = \max_i(v_i) - \min_i(v_i)$.

Ainsi, la dispersion d'un ensemble de transistors est un nombre sans dimension, compris entre 0 et 1. D'ailleurs, que peut-on dire d'un ensemble de transistors dont la dispersion vaut zéro ?

Réponse :

Si la dipersion d'un ensemble de transitor vaut 0. Alors c'est que la formule qui permet de calculer cette dipersion nous donne un résultat égale à 0. Etant donné que l'on prend le maximum entre $\frac{d_h}{MAXHFE}$ et $\frac{d_v}{MAXVBE}$. Alors on peut facilement entre déduire que le minimum entre les deux est également forcément inférieur au maximum par définition. Sachant que la

dipersion est comprise entre 0 et 1. Alors le maximum est 0, et donc le minimum également. Donc $\frac{d_h}{MAXHFE} = \frac{d_v}{MAXVBE}$. On peut donc conclure en disant que si la dipersion d'un ensemble de transistors vaut 0. Alors on a :

$$max_i(h_i) = min_i(h_i)$$

et

$$max_i(v_i) = min_i(v_i)$$

Donc que le maxmimum hfe et minimum hfe sont égaux. Mais également le maximum vbe et le minimum vbe de cet ensemble. Ainsi tous les transistors de cet ensemble on la même valeur hfe et vbe.

Q1 - Calculez la dispersion de l'ensemble des transistors. Combien obtenez vous ?

Note : Les fichiers **src/*.mod** de notre dépôt contiennent le code des des modèles pour les différentes questions. Des commentaires y sont présents pour venir compléter les propos du rapport qui décrit déjà la démarche complète pour obtenir les solutions.

Dans un premier temps nous proposons l'ensemble, les paramètres et variables suivantes :

```
reset;

option solver gurobi;

set TRANS;

param hfe {TRANS};
param vbe {TRANS};
param MAXHFE;
param MAXVBE;

var min_hfe;
var min_vbe;
var max_hfe;
var max_vbe;
```

On ajoute un reset pour permettre de recharger plus simplement notre model. Puis on indique que l'on souhaite utiliser gurobi comme solver.

Ainsi on y retrouve l'ensemble $TRANS$ qui représente et contient tous les transistors. Ces transistors sont caractérisés par une valeur hfe et vbe. C'est pourquoi nous définisons ces deux paramètres qui viennent enrichir l'ensemble TRANS.

On retrouve également quatres variables qui sont min_hfe, min_vbe, max_hfe et max_vbe. Qui vont nous permettre de stocker les valeurs minimale et maximale de hfe et vbe au fur et à mesure du parcours de l'ensemble des transistors.

Et enfin nous aurons besoin des constantes $MAXHFE$ et $MAXVBE$. On leur affecte la valeur 600 à $MAXHFE$ et 1.0 pour $MAXVBE$ pour dans notre fichier **data/transistors_q1.dat** de notre dépôt :

```
param MAXHFE := 600;
param MAXVBE := 1.0;

param : TRANS : hfe vbe =
T1 369.713399 0.505002
T2 172.445927 0.527908
...
...
T32 242.383656 0.619178;
```

De cette manière, la modélisation du contexte est posée. On retrouve bien l'ensemble de nos transistors avec leur caractéristiques et nos constantes.

Nous allons maintenant devoir donner à AMPL des contraintes pour les variables min_hfe, min_vbe, max_hfe et max_vbe. Effectivement il faut indiquer à AMPL que quelque soit le transistor t de l'ensemble TRANS, la valeur du hfe ou vbe de ce transistor doit respceter les comparaison suivantes :

$$min\_hfe \leq hfe_t$$

$$max\_hfe \geq hfe_t$$

$$min\_vbe \leq vbe_t$$

$$max\_vbe \geq vbe_t$$

où hfe_t et vbe_t sont les valeurs hfe et vbe du transistor $t \in TRANS$.

Ainsi on propose les contraintes suivantes en code AMPL en utilisant des subject to pour les transistors t de l'ensemble $TRANS$:

```
subject to h_min {t in TRANS}:
  min_hfe <= hfe[t];

subject to h_max {t in TRANS}:
  max_hfe >= hfe[t];

subject to v_min {t in TRANS}:
  min_vbe <= vbe[t];

subject to v_max {t in TRANS}:
  max_vbe >= vbe[t];
```

Maintenant que l'on est venu apporter à notre modèle les contraintes pour vérifier que les variables min et max de hfe et vbe sont bien les minimum et maximum sur l'ensemble des transistors. Il nous faut apporter les contraintes qui vont nous permettre d'obtenir la dispersion de l'ensemble des transistors.

Tout d'abord il nous faut trouver dans quelle situation nous allons être pour obtenir la dispersion. Est-ce un problème de maximisation ou minimisation ?
Dans notre cas on sait que la formule prend le maximum entre $\frac{d_h}{MAXHFE}$ et $\frac{d_v}{MAXVBE}$. On peut donc déduire que la dispersion va respecter les deux comparaisons suivantes qui reprenent les deux fractions précédemmentes et les formules de l'énoncé :

$$dispersion \geq \frac{d_h}{MAXHFE}$$

où $d_h = \max_i (h_i) - \min_i (h_i)$

$$dispersion \geq \frac{d_v}{MAXVBE}$$

où $d_v = \max_i (v_i) - \min_i (v_i)$

Ainsi on peut effectuer une minimisation sur la variable dispersion. Qui va venir respecter ses deux contraintes.

Enfin on propose donc le code AMPL suivant qui va venir implémenter dans notre modèle la démarche effectuée précédemment :

```
var dispersion;

subject to disp_h:
  dispersion >= ((max_hfe - min_hfe) / MAXHFE);

subject to disp_v:
   dispersion >= ((max_vbe - min_vbe) / MAXVBE);

minimize disp_min:
  dispersion;
```

On déclare notre variable *dispersion*. Puis on indique nos contraintes pour permettre la minimisation de dispersion à l'aide de subject to. Et on indique que l'on souhaite minimiser la variable *dispersion* à l'aide de la commande minimize.

On lance notre modèle **src/q1.mod** dans AMPL en ce plaçant dans le dosser src de notre dépôt :

```
/src$ ampl q1.mod
Gurobi 8.1.0: optimal solution; objective 0.6519189267
dispersion = 0.651919
```

Gurobi trouve une solution optimale pour une dispersion égale à 0.6519189267.

Vérifons cela à la main. Nous devons identifier les hfe et vbe min et max de l'ensemble des transistors. Le hfe minimum est celui du transistor T_{13} avec 89.400099 et le maximum celui du transistor T_{30} avec 480.551455. Le vbe minimum est celui du transistor T_1 avec 0.505002 et le maximum celui du transistor T_{28} avec 0.699465.

On peut vérifier que AMPL a trouvé les mêmes min et max hfe et vbe à l'aide de notre modèle :

```
ampl: display min_hfe;
min_hfe = 89.4001

ampl: display max_hfe;
max_hfe = 480.551

ampl: display min_vbe;
min_vbe = 0.505002

ampl: display max_vbe;
max_vbe = 0.699465
```

On retrouve bien les mêmes valeurs. Ainsi on peut calculer d_h et d_v :

$$d_h = 480.551455 - 89.400099$$
$$d_h = 391.15135599999996$$

$$d_v = 0.699465 - 0.505002$$
$$d_v = 0.19446300000000005$$

On peut maintenant remplacer d_h et d_v dans notre formule de dispersion :

$$D = \max\left(\frac{d_h}{MAXHFE}, \frac{d_v}{MAXVBE}\right)$$

$$D = \max\left(\frac{391.15135599999996}{600}, \frac{0.19446300000000005}{1}\right)$$

$$D = \max\left(0.6519189266666666, 0.19446300000000005\right)$$

$$D = 0.6519189267$$

Ainsi on retrouve bien la même dispersion que gurobi. On peut donc conclure que notre dispersion est bien la dispersion de l'ensemble des transistors.

Pour voir le fichier du modèle en entier : **src/q1.mod**.

La dispersion est un peu élevée (sans doute à cause quelques transistors de valeurs un peu trop extrêmes). On veut recalculer la dispersion de l'ensemble des transistors, en s'autorisant à exclure quelques transistors (au plus E transistors , où E est égal à 4 pour cette question). On peut donc exclure entre 0 (inclus) et E (inclus) transistors.

Q2 - Quelle dispersion obtenez-vous ? Quels transistors ont été exclus ? Est-ce logique ? La dispersion obtenue pourrait-elle être plus grande qu'à la question précédente ?

Pour pouvoir répondre à cette question à partir de notre modèle de la question 1. Il va être nécessaire d'introduire une méthode qui va nous permettre d'inclure ou d'exclure les caractérisques d'un nombre E de transistor(s) de notre ensemble.

Pour cela nous avons repris la démarche du cours lors de la modélisation du problème d'affection d'un ensemble d'étudiants à des créneaux de surveillance d'une salle (page 60 du support).

Ce que l'on souhaite ici avec nos transistors est affecté ou non le transistor dans l'obtention de nos valeurs hfe et vbe minimum et maximum. Ainsi on propose pour cela une variable binary *selection* qui va prendre les transistors t de notre ensemble $TRANS$. On propose donc la définition suivante en code AMPL d'une telle variable :

```
var selection {t in TRANS} binary;
```

Cette variable a donc le même fonctionnement qu'un tableau. Pour un transistor t donné de notre ensemble, le tableau *selection* nous retourne soit 1, soit 0. En fonction de si l'on souhaite inclure ou exclure le transistor la recherche du minimum et maximum du hfe et vbe de notre ensemble $TRANS$.

Il faut pouvoir indiquer également à notre solver l'on souhaite retirer au plus E transistors. Nous déclarons donc un nouveau paramètre E dans notre modèle :

```
param E;
```

Et donc dans notre fichier **data/transistors_q2.dat**. On vient attributer la valeur que l'on souhaite pour E. Donc ici 4:

```
param MAXHFE := 600;
param MAXVBE := 1.0;
param E   := 4;

param : TRANS : hfe vbe =
T1 369.713399 0.505002
...
T32 242.383656 0.619178;
```

Sachant que E vaut 4 ici alors le tableau *selection* qui ne contient que des 1 et 0 doit respecter la propriété suivante :

$$\sum_{t=1}^{32} selection_t \geq |T| - E$$
$$\sum_{t=1}^{32} selection_t \geq 32 - 4$$
$$\sum_{t=1}^{32} selection_t \geq 28$$

où :

- $E = 4$
- $|T|$ est la cardinalité de l'ensemble des transistors $TRANS$
- $t \in TRANS$
- $selection_t$ est la valeur (1 ou 0) pour le transistor t dans *selection*

On peut donc donner la contrainte suivante qui permet d'imposer un minimum pour la cardinalité de transistors inclus dans le calcul de la dispersion de l'ensemble $TRANS$.On rajoute donc en code AMPL à notre modèle :

```
subject to min_trans:
  sum {t in TRANS} selection[t] >= card(TRANS) - E;
```

Une fois que l'on a défini notre tableau *selection* et donné la contrainte sur le nombre de transistor à exclure dans celui-ci. Il nous faut modifier nos contraintes concernant les comparaisons avec les variables min et max pour hfe et vbe. Effectivement il faut pouvoir modifier celles-ci pour indiquer à AMPL à l'aide de notre tableau *selection*. Si l'on doit considère le transistor t actuellement parcouru dans notre ensemble $TRANS$.

Pour permettre cela on peut utiliser le fait que dans notre tableau *selection*, pour un trasistor t. Le tableau va nous donner soit la valeur 1, soit la valeur 0. En fonction de la répartition actuelle des 1 (inclusion) ou 0 (exclusion) pour les transistors dans *selection*. Sachant que le solver va au plus attribués quatres 0 pour l'ensemble des transistors dans *selection* comme vue précédemment. Ainsi on propose les contraintes suivantes qui sont la modi-

fication direct sur celles du modèle de la première question :

$$min\_hfe \leq selection_t \times hfe_t$$
$$max\_hfe \geq selection_t \times hfe_t$$

$$min\_vbe \leq selection_t \times vbe_t$$
$$max\_vbe \geq selection_t \times vbe_t$$

Ici pour max_hfe et max_vbe, il suffit de multiplier la valeur hfe ou vbe du transistor $t \in TRANS$. Par la valeur $selection_t$ qui est donc 1 ou 0. Dans les deux cas la contrainte sera validée si $selection_t$ vaut 1 alors le maximum hfe ou vbe sera forcément supérieur ou égal. Et si $selection_t$ vaut 0 ce sera également le cas.

Par contre pour min_hfe et min_vbe, dans le cas où $selection_t$ vaut 0. Alors on obtiendrait un cas comme celui-ci :

$$min\_hfe \leq selection_t \times hfe_t$$
$$min\_hfe \leq 0 \times hfe_t$$
$$min\_hfe \leq 0$$

$$min\_vbe \leq selection_t \times vbe_t$$
$$min\_vbe \leq 0 \times vbe_t$$
$$min\_vbe \leq 0$$

Nous sommes donc obliger de traiter ce cas où effectivement on obtient une équation qui oblige min_vbe d'être à inférieur ou égale à 0. Du au $selection_t$ du transistor t. Car ce cas provoque l'impossibilité de pouvoir résoudre/lancer notre modèle.

On propose de faire en sorte que dans le cas de min_hfe et min_vbe. Si

$selection_t$ vaut 0. Alors on rajoute à droite le $MAXHFE$ ou $MAXVBE$ que l'on multiplie avant par $1 - selection_t$. Ce qui va permettre de rajouter ou non en fonction de la valeur de $selection_t$. Ainsi on obtient les contraintes suivantes qui nous permettrons de pouvoir lancer notre modèle :

$$min\_hfe \leq selection_t \times hfe_t + (1 - selection_t) \times MAXHFE$$
$$max\_hfe \geq selection_t \times hfe_t$$

$$min\_vbe \leq selection_t \times vbe_t + 1 - selection_t) \times MAXVBE$$
$$max\_vbe \geq selection_t \times vbe_t$$

On propose donc la modification suivante de notre code AMPL des contraintes précédemment étudiée :

```
subject to h_min {t in TRANS}:
 min_hfe <= selection[t] * hfe[t] + (1 -
    selection[t]) * MAXHFE;

subject to h_max {t in TRANS}:
 max_hfe >= selection[t] * hfe[t];

subject to v_min {t in TRANS}:
 min_vbe <= selection[t] * vbe[t] + (1 -
    selection[t]) * MAXVBE;

subject to v_max {t in TRANS}:
 max_vbe >= selection[t] * vbe[t];
```

On lance notre modèle **src/q2.mod** dans AMPL en ce plaçant dans le dosser src de notre dépôt :

```
/src$ ampl q2.mod
Gurobi 8.1.0: optimal solution; objective 0.5666726417
322 simplex iterations
33 branch-and-cut nodes
plus 8 simplex iterations for intbasis
```

```
dispersion = 0.566673

selection [*] :=
 T1 1  T13 1 T17 1 T20 1 T24 1 T28 1 T31 0  T6 1
T10 1  T14 1 T18 1 T21 1 T25 1 T29 1 T32 1  T7 1
T11 0  T15 1 T19 1 T22 1 T26 1  T3 1  T4 1  T8 1
T12 1  T16 0  T2 1 T23 1 T27 1 T30 0  T5 1  T9 1
;
```

Gurobi trouve une solution optimale qui permet d'obtenir une dispersion égale à 0.5666726417. On peut constater à l'aide du tableau sélection que les transistors exlus sont T_{11}, T_{16}, T_{30} et T_{31}. Si on regarde de plus près ces transistors, on peut constater que ce sont les quatres transistors qui ont leur hfe le plus élevé dans l'ensemble $TRANS$:

```
param : TRANS : hfe vbe type =
...
T11 473.747183 0.620745 pnp
...
T16 477.658861 0.673297 npn
...
T30 480.551455 0.530568 pnp
T31 446.039682 0.584432 pnp
...
```

Dans la suite, on oublie cette possibilité d'exclure E transistors. Comme la dispersion est un peu grande pour vendre tous les transistors, Ornicar voudrait découper le stock de transistors en P paquets de transistors (où P vaut 3), ayant chacun le même nombre de transistors (à 1 près). De plus, Ornicar voudrait que les paquets aient une dispersion la plus faible possible.

Q3 - Proposez un critère permettant de minimiser la dispersion des paquets. Plusieurs réponses sont possibles, soyez précis dans la description de votre critère. Si vous trouvez plusieurs critères, vous pouvez discuter les avantages inconvénients de chacun, mais choisissez en un pour la suite.

Pour pouvoir répondre la problèmatique qui est posée. On propose pour pouvoir minimiser la dispersion des paquets, d'en faire la somme. Et donc de chercher à minimiser la somme des dispersions des paquets.

Ainsi si l'on cherche à minimiser la somme de la dispersion des paquets. Nous allons forcément obtenir la solution optimale qui permet d'obtenir la plus petite somme des dispersions. Et donc d'obtenir une répartition des transistors dans les différents paquets, qui permet d'avoir la plus petite dispersion possible pour chaque paquet en fonction de notre ensemble de transistors.

Q4 - Programmez votre critère. Faites une synthèse de la solution obtenue.

Tout d'abord nous allons avoir besoin d'un paramètre NB_PAQUET qui correspond au nombre de paquets P de transistors. Dans notre problème $P = 3$, nous rajoutons donc $NB\_PAQUETS$ dans le fichier **data/transistors_q4.dat** :

```
param MAXHFE := 600;
param MAXVBE := 1.0;
param NB_PAQUET = 3;

param : TRANS : hfe vbe =
T1 369.713399 0.505002
...
T32 242.383656 0.619178;
```

Ainsi dans notre modèle nous aurons besoin d'ajouter le paramètre NB_PAQUET et un ensemble $PAQUET$:

```
param NB_PAQUET > 0;

set PAQUET = 1 .. NB_PAQUET;
```

l'ensemble $PAQUET$ est donc un ensemble qui contient les valeurs allant de 1 à 3 :

```
ampl: display PAQUET;
set PAQUET := 1 2 3;
```

Comme nous allons calculer la dispersion par paquet. Donc de manière indépendante d'un paquet à l'autre. Nous devons modifier nos paramètres min et max hfe et vbe. Pour que l'on puisse accéder à ces paramètres pour un paquet P donné. Ainsi nous modifions nos paramètres de façon à pouvoir inclure comme clé les numéros de paquet de l'ensemble $PAQUET$:

```
var min_hfe {PAQUET};
var min_vbe {PAQUET};
var max_hfe {PAQUET};
var max_vbe {PAQUET};
```

Il en est de même pour notre variable *dispersion* qui devra désormais pouvoir conserver la dispersion obtenu pour un p PAQUET donné. On modifie donc notre variable comme ceci :

```
var dispersion {PAQUET};
```

Nous proposons maintenant de modifier notre variable *selection* pour la renommer en $affectation$ et lui ajouté en paramètre $PAQUET$ pour permettre d'affecter nos transistors à un paquet. La variable reste en binary :

```
var affectation {TRANS,PAQUET} binary;
```

De cette façon, à l'aide d'un t transistor donné et un p paquet donné.

Nous pourrons attribuer la présence (valeur 1) ou l'absence (valeur 0) d'un transistor dans un paquet donné.
Nous aurons également besoin d'une variable *total_disp*. Qui va nous permettre de stocker la valeur de la somme des dispersions des paquets :

```
var total_disp;
```

Comme nous avons modifier notre variable *selection* en $affectattion$ avec un paramètre en plus. Et que les paramètres min et max hfe et vbe sont maintenant propre à chaque paquet. Il est nécessaire de modifier nos contraintes des concernants les minimum et maximum hfe et vbe. Ainsi nous venons apporter les modifications suivantes sur les contraintes :

$$min\_hfe_p \leq affectation_{[t,p]} \times hfe_t + (1 - affectation_{[t,p]}) \times MAXHFE$$
$$max\_hfe_p \geq affectation_{[t,p]} \times hfe_t$$

$$min\_vbe_p \leq affectation_{[t,p]} \times vbe_t + 1 - affectation_{[t,p]}) \times MAXVBE$$
$$max\_vbe_p \geq affectation_{[t,p]} \times vbe_t$$

Nous modifions donc le code AMPL des contraintes ci-dessus de notre modèle :

```
subject to h_min {t in TRANS, p in PAQUET}:
 min_hfe[p] <= affectation[t,p] * hfe[t] + (1 -
    affectation[t,p]) * MAXHFE;

subject to h_max {t in TRANS, p in PAQUET}:
 max_hfe[p] >= affectation[t,p] * hfe[t];

subject to v_min {t in TRANS, p in PAQUET}:
 min_vbe[p] <= affectation[t,p] * vbe[t] + (1 -
    affectation[t,p]) * MAXVBE;
```

```
subject to v_max {t in TRANS, p in PAQUET}:
  max_vbe[p] >= affectation[t,p] * vbe[t];
```

Il faut maintenant pouvoir indiquer que l'on souhaite qu'un transistor soit présent que dans un seul paquet parmi tous les paquets. Pour cela nous pouvons utiliser notre tableau $affectation$ et effectué la somme des valeurs d'un transistor t dans $affectation$ pour tous les paquets p :

$$\sum_{t=1}^{32}\sum_{p=1}^{3} affectation_{[t,p]} = 1$$

où $p \in PAQUET$ et $t \in TRANS$.

Ainsi nous pouvons définir en code AMPL la contrainte d'unicité d'un transistor t parmi tous les paquets p :

```
subject to affectation_unique_transistor {t in TRANS}:
   sum {p in PAQUET} affectation[t,p] == 1;
```

Et pour venir compléter les contraintes de répartition. Il faut indiquer que l'on souhaite précisement une cardinalité qui permet de diviser en trois paquets (NB_PAQUET dans nos paramètres ou P dans le sujet) notre ensemble de transistors $TRANS$. De façon à ce qu'on retrouve la même cardinalité pour chaque paquet, à 1 transistor près. Nous proposons donc la contrainte de cardinalité suivante à l'aide du tableau $affectation$ dans lequel on effectue la somme des 1 qui indique la présence d'un transistor t dans le paquet p :

$$\left\lfloor \frac{|TRANS|}{P} \right\rfloor \leq \sum_{p=1}^{3}\sum_{t=1}^{32} affectation_{[t,p]} \leq \left\lceil \frac{|TRANS|}{P} \right\rceil$$

$$\left\lfloor \frac{32}{3} \right\rfloor \leq \sum_{p=1}^{3}\sum_{t=1}^{32} affectation_{[t,p]} \leq \left\lceil \frac{32}{3} \right\rceil$$

$$10 \leq \sum_{p=1}^{3} \sum_{t=1}^{32} affectation_{[t,p]} \leq 11$$

On définit ainsi la contrainte ci-dessus en code AMPL dans notre modèle :

```
subject to nb_transistors_par_paquets{p in PAQUET}:
  floor(card(TRANS)/NB_PAQUET) <= sum {t in TRANS}
     affectation[t,p] <= ceil(card(TRANS)/NB_PAQUET);
```

Avant de pouvoir lancer notre modèle, il nous reste encore à modifier les contraintes qui portent sur la variable *dispersion*. Qui dépend maintenant d'un paramètre p qui est le numéro du paquet. Mais également de définir une contrainte pour pouvoir permettre de minimiser notre objectif. Qui est la somme des dispersions des paquets.

Ce qui change pour les contraintes qui portent sur la variable *dispersion* est l'ajout du paramètre $p \in PAQUET$. Il nous suffit donc d'ajouter cette considération :

$$dispersion_p \geq \frac{d_{h_p}}{MAXHFE}$$

où $d_{h_p} = max\_hfe_p - min\_hfe_p$

$$dispersion_p \geq \frac{d_{v_p}}{MAXVBE}$$

où $d_{v_p} = max\_vbe_p - min\_vbe_p$

Voici donc en code AMPL les deux contraintes ci-dessus, que l'on modifie dans notre modèle :

```
subject to disp_h {p in PAQUET}:
  dispersion[p] >= ((max_hfe[p] - min_hfe[p]) /
     MAXHFE);
```

```
subject to disp_v {p in PAQUET}:
  dispersion[p] >= ((max_vbe[p] - min_vbe[p]) /
    MAXVBE);
```

Concernant la contrainte de la somme des dipsersions des paquets. Comme allons vouloir la minimiser il nous faut déclarer de cette manière une telle contrainte :

$$total\_disp \geq \sum_{p=1}^{3} dispersion_p$$

Ce que l'on ajoute dans notre modèle en code AMPL comme suit :

```
subject to sum_disp:
  total_disp >= sum {p in PAQUET} dispersion[p];
```

Il ne nous reste plus qu'à modifier notre objectif qui est maintenant. De minimiser la variable *total_disp*. Donc la somme des dispersions parmi tous les paquets :

```
minimize disp_min:
  total_disp;
```

Ainsi nous lançons notre modèle **src/q4.mod** depuis le dossier **src/** de notre dépôt :

```
/src$ ampl q4.mod
Gurobi 8.1.0: optimal solution; objective 0.5818244
8828 simplex iterations
566 branch-and-cut nodes
plus 29 simplex iterations for intbasis
total_disp = 0.581824

dispersion [*] :=
1 0.249314
```

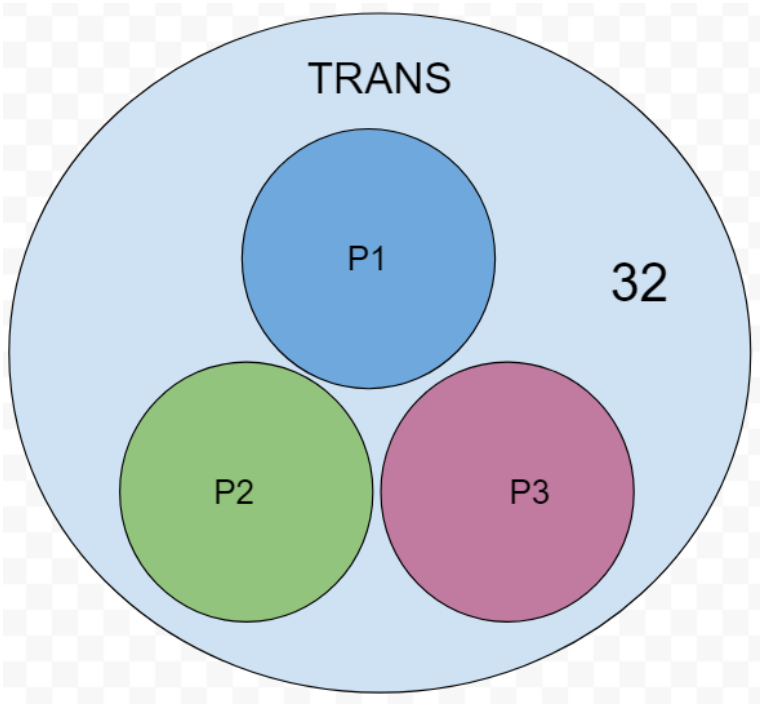
```
2 0.165978
3 0.166533
;

affectation [*,*]
:    1  2  3   :=
T1   1  0  0
T10  0  1  0
T11  0  0  1
T12  0  0  1
T13  0  1  0
T14  1  0  0
T15  0  0  1
T16  0  0  1
T17  0  1  0
T18  0  1  0
T19  0  0  1
T2   0  1  0
T20  1  0  0
T21  0  1  0
T22  0  0  1
T23  0  1  0
T24  1  0  0
T25  0  1  0
T26  0  1  0
T27  0  0  1
T28  1  0  0
T29  1  0  0
T3   0  0  1
T30  0  0  1
T31  0  0  1
T32  1  0  0
T4   1  0  0
T5   0  1  0
T6   1  0  0
T7   0  1  0
T8   0  0  1
T9   1  0  0
```

```
;
```

Ainsi on peut voir que gurobi trouve une solution optimale pour une somme totale des dispersions des paquets de 0.5818244. On peut voir que le paquet 1 à une dispersion de 0.249314, le paquet 2 de 0.165978 et le paquet 3 de 0.166533. Nous pouvons donc constater que ce sont des dipserions plus basses, plus intéressantes pour prétendre à une vente de nos paquets de transistors. A comparer une dispersion de l'ensemble des transistors comme lors du premier ou deuxième modèle.

On peut vérifier également au travers du tableau $affectation$. Que la répartation, ou l'affectation des transistors pour reprendre le nom de la variable. Respecte effectivement les différentes contraintes que nous avons imposés. A savoir qu'aucun transistor est présent dans deux paquets différents. Et qu'un paquet contient - dans le contexte de notre exercice sinon voir formulation générale - au moins 10 transistors et au plus 11 transistors. Ce qui va nous permettre d'obtenir une division de notre ensemble comme ceci :



Ainsi, deux ensembles des trois sous ensembles de $TRANS$ aurons une

cardinalité égale à 11. Et le paquet restant une cardinalité de 10. Comme l'ensemble $TRANS$ est composé d'exactement 32 transistors et que l'on exclut aucun transistor pour cet exercice.

Ornicar n'est pas très convaincu du découpage en paquets, et vous propose une autre idée. Cette idée consiste à trouver des paires de transistors de caractéristiques proches. On appelle cela appairer des transistors (on parle aussi d'appariement de transistors). Pour savoir si une paire de transistors est bien appairée, on va simplement considérer la dispersion de l'ensemble formé de ces deux transistors. De plus, pour des raisons techniques, Ornicar ne veut pas considérer de paires de transistors dont la dispersion est supérieure ou égale à 0.12. On ne peut appairer que des transistors de type différent. Pour terminer, Ornicar a conscience que certains transistors risquent de ne pas être appairés à cause des contraintes précédentes. Toutefois, Ornicar souhaite en priorité un grand nombre de paires de transistors, la dispersion faible est un critère moins prioritaire.

Q5 - Proposez un critère pour exprimer qu'on a obtenu des paires de transistors les mieux appairées possibles. Programmez votre critère, et donnez une synthèse de votre solution.

Nous avons tenter plusieurs approches en repartant de notre modèle de la question précédente. Mais nous avons été confrontés à différent obstacle comme un modèle non linéaire, un modèle contenant trop de variables. Nous avons donc fait le choix de reprendre le problème d'indépendamment des problématiques rencontrées précédemment. Tout en gardant en tête les points clés de nos modélisations, qui nous ont permis la recherche des solutions. Voici donc la démarche que nous avons eu pour permettre de répondre aux attentes d'Orinicar.

Dans ce problème, Ornicar souhaite obtenir le maximum de pairs de transistors qui peuvent être appairées. C'est-à-dire que chaque pairs contient un transistor du type *npn* et *pnp*. Et que la dispersion calculée entre les deux transistors de la pair soit inférieur à 0.12.

Nous nous sommes donc poser la question en partant de cela. Comment

pouvons nous en utilisant le moins de variable possible, définir un critère qui permet de vérifier et obtenir un maximum de pairs qui respecte une valeur de disperison.

Dans un premier temps, nous ne pouvons plus considérer l'ensemble de nos transistors $TRANS$. Comme ensemble sur lequel nous pourrons directement travailler. Dans le sens où, comme il faut maintenant pouvoir différencier les transistors de type *pnp* et *npn* pour former des pairs. Il n'est pas nécessaire d'effectuer des contraintes ou définir des variables qui contiendra des pairs de transistors qui ne sont pas valides de par leur différence de type. On propose donc d'ajouter dans notre fichier de données **data/transistors_q5.dat** les paramètres MAX_DISP, pnp_t et npn_t :

```
param MAXHFE := 600;
param MAXVBE := 1.0;
param MAX_DISP := 0.12;
param pnp_t = pnp;
param npn_t = npn;

param : TRANS : hfe vbe type =
T1 369.713399 0.505002 npn
...
...
T32 242.383656 0.619178 npn;
```

Ainsi nous allons pouvoir définir les paramètres et les deux sous ensembles de $TRANS$ suivant dans notre modèle :

```
set TRANS;

param hfe {TRANS};
param vbe {TRANS};
param type {TRANS} symbolic;

param MAXHFE;
```

```
param MAXVBE;

param MAX_DISP;
param pnp_t symbolic;
param npn_t symbolic;

set PNP within {TRANS} := {t in TRANS : type[t] ==
   pnp_t};
set NPN within {TRANS} := {t in TRANS : type[t] ==
   npn_t};
```

On utilise donc les paramètres pnp_t et npn_t pour pouvoir à l'aide de la notation *within* et l'opérateur " : ". Pour pouvoir indiquer que l'on retrouve des éléments de $TRANS$ dans notre sous ensemble. Et que l'on condition leur présence en fonction de leur paramètre *type*. Ainsi on obtient les deux sous ensembles de types de transistors de $TRANS$:

```
ampl: display PNP;
set PNP := T2 T3 T4 T5 T7 T11 T14 T15 T19 T20 T21 T22
   T28 T29 T30 T31;

ampl: display NPN;
set NPN := T1 T6 T8 T9 T10 T12 T13 T16 T17 T18 T23
   T24 T25 T26 T27 T32;
```

De cette façon nous pourrons construitre une variable $affectation$ qui contiendra une valeur binaire pour tout couple (t_1, t_2). Où $t_1 \in PNP$ et $t_2 \in NPN$. On déclare donc une variable binary $affectation$ dans notre modèle :

```
var affectation{p in PNP,n in NPN} binary;
```

On retrouve donc un tableau avec le format suivant :

```
ampl: display affectation;
affectation [*,*]
```

```
:   T1 T10 T12 T13 T16 T17 T18 T23 T24 T25 T26 T27
   T32 T6 T8 T9 :=
T11  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T14  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T15  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T19  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T2   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T20  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T21  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T22  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T28  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T29  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T3   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T30  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T31  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T4   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T5   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T7   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
;
```

Ainsi la valeur 1 indiquera que le couple (t_1, t_2) forme une pair de dispersion valide et qui permet d'obtenir par association de ceux-ci. Un nombre de pairs possibles maximum. Ce qui est que l'on cherche à faire. Et la valeur 0 que le couple (t_1, t_2) n'est soit pas une pair de transistors qui ont une dispersion valide. Ou qui ne permet pas de pouvoir appairer un maximum de pairs de par leur association en pair.

Comme nous cherchons seulement à avoir des dipsersions inférieur à 0.12. Et que la minimisation des dispersions des pairs de transistors est secondaire face à la recherche de former un maximum de pair. On définir notre variable d'objectif $total\_affectation$:

```
var total_affectation >= 0;
```

Ainsi nous avons former notre critère au travers de cette variable et de la définition de notre tableau $affectation$. Qui sera donc la maximisation de la

variable d'objectif qui elle sera le nombre de 1 dans le tableau $affectation$. Donc du nombre possible de pairs de transistors qui respectent une borne supérieur de dispersion égale à 0.12. Et qui par affectation/association permettent en priorité la maximisation de la variable d'objectif.

Ceci étant posé. Nous avons procédé par étape ensuite en exprimant directement des contraintes portant sur ce que l'on souhaite retrouver dans notre tableau $affectation$. La première étant que si un transistor PNP forme une pair avec un transistor NPN. Alors il ne doit être présent dans aucune autre pair pour la répartition actuelle du tableau $affectation$. Comme $affectation$ contient des valeurs binaires on peut décrire cela par :

$$\sum_{p_1}^{p_x} \sum_{n_1}^{n_y} affectation_{[p,n]} \leq 1$$

où $p_1, ..., p_x \in PNP$, $n_1, ..., n_y \in NPN$, $x = |PNP|$ avec $y = |NPN|$

Effectivement on rappelle qu'il est possible que des transistors ne puisse être présent dans une pair. C'est pourquoi cette somme doit être inférieure ou égale à 1. Ainsi en code AMPL on peut donc définir la contrainte suivante dans notre modèle :

```
subject to affectation_unique_PNP{p in PNP}:
  sum {n in NPN} affectation[p,n] <= 1;
```

Cette contrainte va aussi s'appliquer aux colonnes de notre tableau $affectation$. Qui sera donc la même contrainte mais pour les transistors de NPN. Ainsi l'on vient former la contrainte suivante :

$$\sum_{n_1}^{n_y} \sum_{p_1}^{p_x} affectation_{[p,n]} \leq 1$$

On vient donc compléter notre modèle par le code AMPL qui définit cette contrainte :

```
subject to affectation_unique_NPN {n in NPN}:
  sum {p in PNP} affectation[p,n] <= 1;
```

Maintenant que nous avons exprimer les contraintes qui vont nous permettre d'interdire la présence des transistors dans plusieurs pairs. Il faut faire vérifier à notre modèle que s'il forme un couple (t_1, t_2). Donc qu'il souhaite affecté dans notre tableau $affectation$ ce couple de transistors en pair alors la dispersion de celle-ci doit respecter la borne supérieure, la limite que l'on impose en tant que dispersion acceptable pour une pair de transistors. Donc dans notre cas cette limite est MAX_DISP. Mais il faut aussi formuler notre contrainte pour généraliser sur notre tableau $affectation$, c'est-à-dire que cette contraitre doit être vérifier également dans le cas où on n'affecte pas nos deux transistors en tant que pair.

Comme $affectation$ est un tableau binaire, nous pouvons donc multipler la dispersion de la pair de transistors par la valeur contenu dans le tableau $affectation$ pour cette même pair. Ainsi soit on affecte les deux transistors en pair et alors on multiplie par 1. Soit ce n'est pas le cas et on multiplie par 0. Et ainsi dans les deux cas la dispersion doit être inférieur à MAX_DISP. Nous pouvons donc formuler notre contrainte de la sorte :

$$affectation_{[p,n]} \times \max\left(\frac{|vbe_p - vbe_n|}{MAXVBE}, \frac{|hfe_p - hfe_n|}{MAXHFE}\right) \leq MAX\_DISP$$

où $\forall p \in PNP, \forall n \in NPN$ et $MAX\_DISP = 0.12$

A noter qu'ici nous prenons la valeur absolue des soustractions des hfe et vbe. Comme nous devons pas trouver de minimum et maximum hfe et vbe. Etant donné que l'on a des pairs de transistors. On peut donc traduire une telle contrainte en code AMPL dans notre modèle de la façon suivante :

```
subject to affect_valide_disp{p in PNP, n in NPN}:
  affectation[p,n] * (max(abs(vbe[p] -
     vbe[n])/MAXVBE,abs(hfe[p] - hfe[n])/MAXHFE)) <=
     MAX_DISP;
```

Ainsi il nous reste maintenant les contraintes portant que l'objectif et son lien avec le tableau $affectation$ et nos ensembles PNP et NPN. Effectivement notre objectif premier est de permettre la maximisation du nombre de pairs possibles ayant une dispersion valide. Notre variable d'objectif étant $total\_affectation$, et comme nous l'avons mentionner au début de notre démarche. Notre critère est donc d'avoir le maximum de valeur à 1 dans notre tableau $affectation$. Qui à ce stade du modèle va correspondre à la validation de la dispersion entre une pair de transistors. Et que leur appairage permet de faire en sorte que l'on obtient un nombre maximum d'autres pairs de dispersion valide qui respectent également ce dernier critère. On peut donc définir la contrainte suivante pour pouvoir maximiser notre variable d'objectif :

$$total\_affectation \leq \sum_{p} \sum_{n} affectation_{[p,n]}$$
$$\forall p \in PNP, \forall n \in NPN$$

De cette façon on obtient la somme de tous les 1 dans le tableau $affectation$ et donc le nombre de pair possible qui permet de répondre aux attentes de Ornicar. On peut donc traduire cette contrainte en code AMPL suivant dans notre modèle :

```
subject to sum_affectation:
  total_affectation <= sum {p in PNP, n in NPN}
     affectation[p,n];
```

Ainsi l'on peut noter à titre informatif, est que notre variable d'objectif est bornée par :

$$0 \leq total\_affectation \leq \min\left(|PNP|, |NPN|\right)$$

Et enfin comme nous cherchons à maximiser notre variable d'objectif. Alors nous ajoutons à notre modèle le code AMPL qui le permet :

```
maximize nb_pair:
  total_affectation;
```

Ainsi on lance notre modèle **src/q5.mod** depuis le dossier **src/** de notre dépôt :

```
/src$ ampl q5.mod
Gurobi 8.1.0: optimal solution; objective 12
22 simplex iterations
plus 1 simplex iteration for intbasis
total_affectation = 12

affectation [*,*]
:   T1 T10 T12 T13 T16 T17 T18 T23 T24 T25 T26 T27
   T32 T6 T8 T9 :=
T11  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T14  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
T15  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T19  0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
T2   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0
T20  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T21  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0
T22  0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0
T28  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0
T29  0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0
T3   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T30  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
T31  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0
T4   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T5   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
T7   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
;
```

Gurobi trouve donc un objectif avec 12 pairs possibles de transistors. Que l'on peut retrouver grâce au tableau $affectation$. On voit donc qu'effectivement il y a un seul 1 par ligne et par colonne dans le tableau $affectation$. Nous allons maintenant vérifier que toutes les pairs que gurobi nous indique comme avec une dispersion valide, le sont réellement. Pour cela nous proposons une variable calculé $dispersion$. Qui va nous donner les dispersions de toutes les pairs possibles du tableau $affectation$. On propose donc la variaible suivant :

```
var dispersion_flat{p in PNP, n in NPN}:=
  max(abs(vbe[p] - vbe[n])/MAXVBE,abs(hfe[p] -
     hfe[n])/MAXHFE);
```

Ici on itère donc sur tous les $p \in PNP$ et tous les $n \in NPN$. Pour attribuer à $dispersion_{[p,n]}$ la dispersion calculé entre les deux transistors p et n. L'affichage de ce tableau est disponible en dernière page de ce rapport (page 37) dans la section 3 Données. Ainsi on retrouve les dispersions de nos pairs que gurobi a sélectionnés pour obtenir un maximum de pairs possible tout en respectant la limite MAX_DISP :

$$D_{(T_{14},T_6)} = 0.0134226 \qquad D_{(T_{19},T_8)} = 0.104778$$
$$D_{(T_2,T_{17})} = 0.099229 \qquad D_{(T_{21},T_{23})} = 0.069471$$
$$D_{(T_{22},T_{27})} = 0.082883 \qquad D_{(T_{28},T_{32})} = 0.080287$$
$$D_{(T_{29},T_{24})} = 0.0903013 \qquad D_{(T_{30},T_{12})} = 0.109476$$
$$D_{(T_{31},T_{16})} = 0.088865 \qquad D_{(T_4,T_1)} = 0.0407786$$
$$D_{(T_5,T_{13})} = 0.116531 \qquad D_{(T_7,T_9)} = 0.0998416$$

Ainsi on peut confirmer que gurobi a pu former des pairs de transistors avec une dispersion valide. Avec la présence des transistors dans une seule pairs. Concernant la maximisation du nombre de pairs possibles, donc de notre variable $total\_affectation$. Le nombre de pairs est de 12. Ce qui est très proche des 16 pairs possibles sans la contrainte de la dispersion qui doit être inférieur à 0.12. On peut donc conclure en disant que notre modèle est cohérent et qu'il nous permet bien de modéliser la demande de Ornicar.

Q6 - Si votre modèle ne comporte pas trop de contraintes ni de variables, vous devriez pouvoir appairer tous les transistors en augmentant un peu la contrainte de dispersion de 0.12 imposée sur les paires. Quelle dispersion minimale permet d'appairer tous les transistors ? Justifiez votre raisonnement.

Pour permettre cela il nous suffit de repartir de notre modèle **src/q5.mod**. En modifiant quelques éléments.
Dans un premier temps nous devons changer notre objectif premier qui est la recherche par minimisation de la dispersion qui puisse permettre d'apparier tous les transistors. Ainsi nous ajoutons la définition la variable calculé *disp* :

```
var disp;
```

Puis il faut indiquer à notre modèle l'encadrement de notre variable *disp*. Une dispersion est une valeur comprise entre 0 et 1. Ainsi on définit une contrainte pour cela :

$$0 \leq disp \leq 1$$

Ce qu'on traduit en code AMPL dans notre modèle de la manière suivante à l'aide d'une contrainte :

```
subject to disp_borne:
   0 <= disp <= 1;
```

Ensuite il nous fait indiquer à notre modèle que l'on souhaite maintenant obtenir un nombre, dans notre variable $total\_affectation$. Qui soit également au nombre possible de pairs de trasistors de type NPN et PNP possibles. Sans considération de dispersion ici, ça sera le rôle de notre variable d'objectif. Nous pouvons donc formuler une contrainte qui impose cela dans notre modèle :

$$total\_affection = \min(|PNP|, |NPN|)$$

Ici on prend le minimum entre la cardinalité de l'ensemble PNP et NPN. Car effectivement si l'un des deux ensembles a une cardinalité moins élevée que l'autre. Alors il sera possible au plus de faire autant de pairs qu'il y a de transistors dans l'ensemble du type de cardinalité la plus faible. Dans notre cas, cela ne pose pas de soucis étant donner que les deux ensembles on autant de transistors. Mais nous avons fait ce choix pour généraliser notre démarche et considérer ce cas. On peut donc traduire cette contrainte en code AMPL dans notre modèle :

```
subject to total_affectation_must_max_possible_peer:
  total_affectation == min(card(PNP),card(NPN));
```

Il faut maintenant modifier l'objectif de notre modèle qui est maintenant la minimisation de notre variable *disp* :

```
minimize dispersion_for_maximum_peer:
  disp;
```

Nous lançons donc notre modèle **src/q6.mod** :

```
/src$ ampl q6.mod
Gurobi 8.1.0: optimal solution; objective 0.2494821183
1379 simplex iterations
1 branch-and-cut nodes
plus 1 simplex iteration for intbasis
disp = 0.249482

total_affectation = 16

affectation [*,*]
:   T1 T10 T12 T13 T16 T17 T18 T23 T24 T25 T26 T27
   T32 T6 T8 T9 :=
T11  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0
T14  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0
T15  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0
```

```
T19  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
T2   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
T20  1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T21  0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0
T22  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
T28  0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0
T29  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0
T3   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0
T30  0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
T31  0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0
T4   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
T5   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
T7   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
;
```

On peut voir que le $total\_affection$ est égale à 16. Ainsi Gurobi trouve une solution optimale pour un intervalle de dispersion acceptées comprises entre 0 et 0.2494821183, qui permet d'appairer tous les transistors.

2 - Modèles

Vous pouvez retrouver les modèles dans les fichiers **q1.mod**, **q2.mod**, **q4.mod**, **q5.mod** et **q6.mod** du dossier **src/** de notre dépôt.

3 - Données

Vous pouvez retrouver les données dans les fichiers **transistors_q1.dat**, **transistors_q2.dat**, **transistors_q4.dat**, **transistors_q5.dat** et **transistors_q6.dat** du dossier **data/** de notre dépôt.

Voici les données de départ contenu dans le fichier **transistors_q1.dat** :

```
param MAXHFE := 600;
param MAXVBE := 1.0;
```

```
param : TRANS : hfe vbe type =
T1 369.713399 0.505002 npn
T2 172.445927 0.527908 pnp
T3 388.349744 0.678436 pnp
T4 345.246256 0.506357 pnp
T5 159.318987 0.601071 pnp
T6 330.622531 0.643204 npn
T7 160.220311 0.617853 pnp
T8 429.403684 0.531932 npn
T9 220.125258 0.531096 npn
T10 101.105138 0.575985 npn
T11 473.747183 0.620745 pnp
T12 414.865893 0.607246 npn
T13 89.400099 0.558636 npn
T14 338.676073 0.640914 pnp
T15 380.631659 0.654614 pnp
T16 477.658861 0.673297 npn
T17 188.986802 0.627137 npn
T18 131.327049 0.571054 npn
T19 400.910157 0.636710 pnp
T20 354.565503 0.534228 pnp
T21 131.701247 0.575891 pnp
T22 392.307125 0.668570 pnp
T23 164.524036 0.506420 npn
T24 250.582394 0.513238 npn
T25 156.313272 0.599846 npn
T26 121.435795 0.527926 npn
T27 423.506906 0.585687 npn
T28 230.998222 0.699465 pnp
T29 304.763147 0.518182 pnp
T30 480.551455 0.530568 pnp
T31 446.039682 0.584432 pnp
T32 242.383656 0.619178 npn;
```

Voici le contenu de la variable *dispersion* du modèle **src/q5.mod** :

```
/src$ ampl
ampl: model q5.mod
Gurobi 8.1.0: optimal solution; objective 12
22 simplex iterations
plus 1 simplex iteration for intbasis
...
...
ampl: display dispersion;
dispersion [*,*]
```

Voir image page suivante...

```
:        T1          T10         T12         T13         T16         T17       :=
T11    0.17339     0.62107     0.0981355   0.640578    0.052552    0.474601
T14    0.135912    0.395952    0.126983    0.41546     0.231638    0.249482
T15    0.149612    0.465878    0.0570571   0.485386    0.161712    0.319408
T19    0.131708    0.499675    0.029464    0.519183    0.127915    0.353206
T2     0.328779    0.118901    0.404033    0.13841     0.508688    0.099229
T20    0.029226    0.422434    0.100501    0.441942    0.205156    0.275965
T21    0.396687    0.0509935   0.471941    0.0705019   0.576596    0.0954759
T22    0.163568    0.485337    0.061324    0.504845    0.142253    0.338867
T28    0.231192    0.216488    0.306446    0.235997    0.411101    0.072328
T29    0.10825     0.33943     0.183505    0.358938    0.28816     0.192961
T3     0.173434    0.478741    0.07119     0.498249    0.148849    0.332272
T30    0.18473     0.632411    0.109476    0.651919    0.142729    0.485941
T31    0.12721     0.574891    0.0519563   0.594399    0.088865    0.428421
T4     0.0407786   0.406902    0.116033    0.42641     0.220688    0.260432
T5     0.350657    0.0970231   0.425912    0.116531    0.530566    0.0494464
T7     0.349155    0.0985253   0.424409    0.118034    0.529064    0.0479442

:        T18         T23         T24         T25         T26         T27       :=
T11    0.5707      0.515372    0.371941    0.529057    0.587186    0.0837338
T14    0.345582    0.290253    0.146823    0.303938    0.362067    0.141385
T15    0.415508    0.360179    0.216749    0.373864    0.431993    0.0714587
T19    0.449305    0.393977    0.250546    0.407661    0.465791    0.051023
T2     0.0685315   0.021488    0.130227    0.071938    0.0850169   0.418435
T20    0.372064    0.316736    0.173305    0.33042     0.38855     0.114902
T21    0.004837    0.069471    0.198135    0.04102     0.047965    0.486343
T22    0.434967    0.379638    0.236208    0.393323    0.451452    0.082883
T28    0.166119    0.193045    0.186227    0.124475    0.182604    0.320848
T29    0.28906     0.233732    0.0903013   0.247416    0.305546    0.197906
T3     0.428371    0.373043    0.229612    0.386727    0.444857    0.092749
T30    0.582041    0.526712    0.383282    0.540397    0.598526    0.0950742
T31    0.524521    0.469193    0.325762    0.482877    0.541006    0.0375546
T4     0.356532    0.301204    0.157773    0.314888    0.373017    0.130434
T5     0.0466532   0.094651    0.152106    0.00500952  0.073145    0.440313
T7     0.0481554   0.111433    0.150603    0.018007    0.089927    0.438811

:        T32         T6          T8          T9        :=
T11    0.385606    0.238541    0.088813    0.422703
T14    0.160487    0.0134226   0.151213    0.197585
T15    0.230413    0.0833485   0.122682    0.267511
T19    0.264211    0.117146    0.104778    0.301308
T2     0.116563    0.263628    0.428263    0.0794656
T20    0.18697     0.108976    0.12473     0.224067
T21    0.184471    0.331535    0.496171    0.147373
T22    0.249872    0.102808    0.136638    0.28697
T28    0.080287    0.166041    0.330676    0.168369
T29    0.103966    0.125022    0.207734    0.141063
T3     0.243277    0.096212    0.146504    0.280374
T30    0.396946    0.249882    0.0852463   0.434044
T31    0.339427    0.192362    0.0525      0.376524
T4     0.171438    0.136847    0.140262    0.208535
T5     0.138441    0.285506    0.450141    0.101344
T7     0.136939    0.284004    0.448639    0.0998416
;
```