

# Rapport Algorithmique 3

Zia Gulzar 000595624  
Ando Rasamimanana 000570596

2025-2026

## 1 Introduction

Des tournées de livraison logistique à la conception de microprocesseurs, l'optimisation des trajets constitue un enjeu économique et industriel majeur. Cette réalité concrète se modélise par le célèbre problème du voyageur de commerce (TSP), dont la résolution exacte se heurte rapidement à une complexité exponentielle. Au cœur de ce défi algorithmique se trouve la question de l'élimination des sous-tours. Ce rapport propose une analyse comparative entre deux méthodes historiques : la formulation compacte de Miller-Tucker-Zemlin (MTZ), économe en contraintes ( $\mathcal{O}(n^2)$ ), et l'approche de Dantzig-Fulkerson-Johnson (DFJ), qui mobilise un nombre exponentiel d'inégalités ( $\mathcal{O}(2^n)$ ). Notre objectif est de comprendre pourquoi, contre toute attente, la méthode la plus lourde théoriquement (DFJ) s'avère être l'outil le plus performant pour répondre aux exigences d'efficacité du monde réel.

## 2 Mécanismes d'exclusion des sous-tours

Soit une instance de  $n = 5$  villes dont l'ensemble des sommets est  $V = \{0, 1, 2, 3, 4\}$ . Supposons qu'une solution contienne deux sous-tours disjoints : un cycle  $(1 \rightarrow 2 \rightarrow 3 \rightarrow 1)$  et un cycle  $(0 \rightarrow 4 \rightarrow 0)$ .

### 2.1 Approche MTZ

La formulation MTZ repose sur l'introduction de variables continues  $u_i$  pour  $i \in \{1, \dots, n-1\}$ , représentant l'ordre de passage au sommet  $i$ . La contrainte d'exclusion est définie par :

$$u_i - u_j + n \cdot x_{ij} \leq n - 1, \quad \forall (i, j) \in V \setminus \{0\}, i \neq j \quad (1)$$

Pour le sous-tour  $(1 \rightarrow 2 \rightarrow 3 \rightarrow 1)$ , les variables binaires valent  $x_{12} = 1, x_{23} = 1$  et  $x_{31} = 1$ . En appliquant la contrainte avec  $n = 5$ , nous obtenons le système d'inégalités suivant :

- Pour l'arc  $1 \rightarrow 2$  :  $u_1 - u_2 + 5(1) \leq 4 \implies u_1 + 1 \leq u_2$
- Pour l'arc  $2 \rightarrow 3$  :  $u_2 - u_3 + 5(1) \leq 4 \implies u_2 + 1 \leq u_3$
- Pour l'arc  $3 \rightarrow 1$  :  $u_3 - u_1 + 5(1) \leq 4 \implies u_3 + 1 \leq u_1$

Par sommation de ces inégalités, nous obtenons  $u_1 + u_2 + u_3 + 3 \leq u_1 + u_2 + u_3$ , ce qui simplifie à  $3 \leq 0$ . Cette contradiction prouve que le cycle ne peut pas exister.

*Note : Le second sous-tour  $(0 \rightarrow 4 \rightarrow 0)$  est autorisé par ces équations spécifiques car elles ne s'appliquent pas au sommet 0. Cependant, les contraintes de flux (degré entrant et sortant égal à 1) forceront le sommet 0 à être relié au reste du graphe, rendant ce sous-tour impossible dans une solution globale valide.*

### 2.2 Approche DFJ

La formulation DFJ interdit les sous-tours en limitant le nombre d'arcs à l'intérieur de tout sous-ensemble de sommets  $S \subset V$  (où  $2 \leq |S| \leq n-1$ ). La contrainte s'énonce ainsi :

$$\sum_{i \in S} \sum_{j \in S, j \neq i} x_{ij} \leq |S| - 1 \quad (2)$$

Considérons le sous-ensemble  $S = \{1, 2, 3\}$ , dont le cardinal est  $|S| = 3$ . Si le sous-tour  $(1 \rightarrow 2 \rightarrow 3 \rightarrow 1)$  est présent, les variables d'arcs internes au sous-ensemble valent :

$$x_{12} = 1, \quad x_{23} = 1, \quad x_{31} = 1 \quad (3)$$

Le calcul de la somme des arcs donne :

$$\sum_{i, j \in S} x_{ij} = x_{12} + x_{23} + x_{31} = 3 \quad (4)$$

Or, la contrainte impose que cette somme soit inférieure ou égale à  $|S| - 1 = 2$ . Nous obtenons  $3 \leq 2$ , ce qui constitue une violation de contrainte. Le solveur rejette donc cette solution.

### 3 Comparaison des formulations MTZ et DFJ

Cette section se consacre à l'étude comparative des deux formulations implémentées : Miller-Tucker-Zemlin (MTZ) et Dantzig-Fulkerson-Johnson (DFJ).

#### 3.1 Résultats Expérimentaux

Le tableau ci-dessous présente une comparaison exhaustive des performances entre la formulation compacte (MTZ), la résolution par génération itérative (DFJ Itératif) et l'énumération complète (DFJ Énumératif).

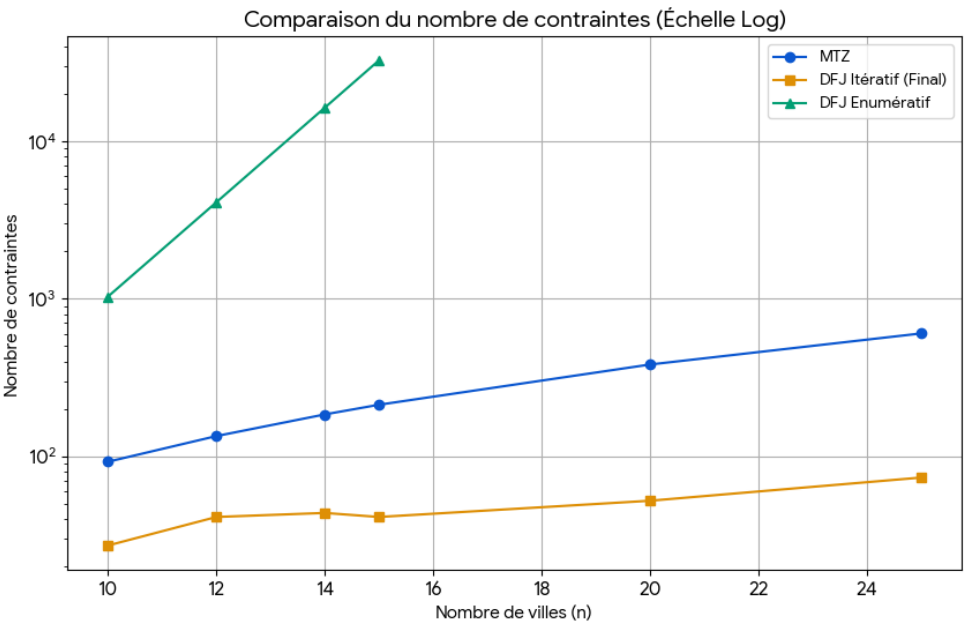


FIGURE 1 – Comparaison du nombre de contraintes générées par MTZ, DFJ Itératif et DFJ Énumératif

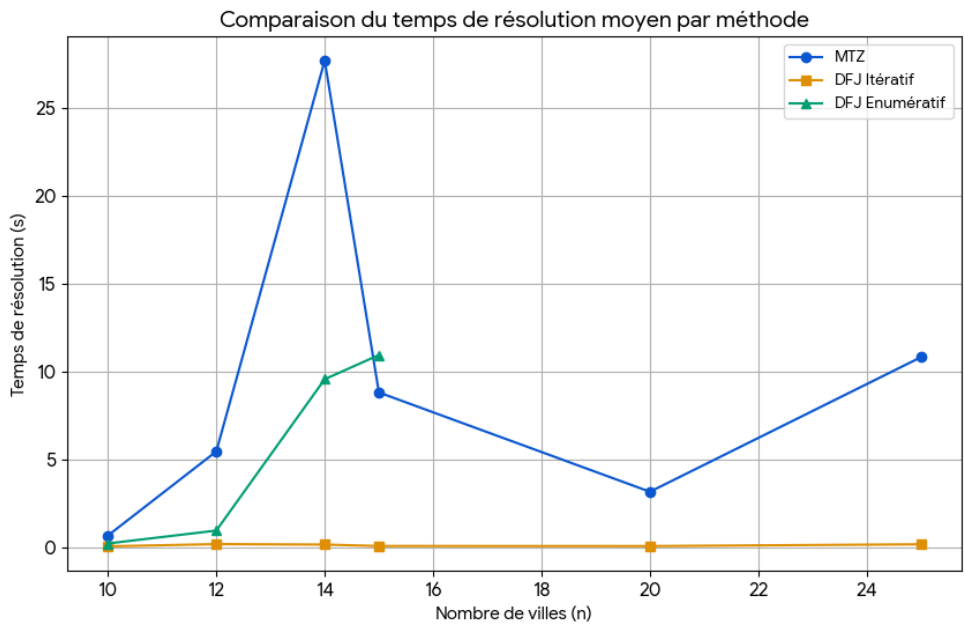


FIGURE 2 – Comparaison des temps de résolution selon la formulation et la taille des instances

Inst.	n	MTZ				DFJ Itératif				DFJ Enum			
		Obj	T(s)	V	C	Obj	T(s)	V	C	Obj	T(s)	V	C
10_circle_1	10	298.6	0.39	99	92	298.6	0.12	90	28	298.6	0.32	90	1032
10_euclidean_1	10	207.8	0.40	99	92	207.8	0.03	90	25	207.8	0.19	90	1032
10_euclidean_2	10	323.9	0.29	99	92	323.9	0.04	90	28	323.9	0.17	90	1032
10_line_1	10	156.1	2.16	99	92	156.1	0.07	90	33	156.1	0.16	90	1032
10_random_asym_1	10	191.5	0.05	99	92	191.5	0.01	90	20	191.5	0.22	90	1032
10_random_sym_1	10	281.0	0.73	99	92	281.0	0.05	90	28	281.0	0.26	90	1032
12_circle_1	12	265.2	2.85	143	134	265.2	0.08	132	39	265.2	0.91	132	4106
12_line_1	12	174.8	8.05	143	134	174.8	0.28	132	43	174.8	1.00	132	4106
14_circle_1	14	304.5	3.07	195	184	304.5	0.03	182	34	304.5	5.70	182	16396
14_line_1	14	188.0	52.27	195	184	188.0	0.28	182	53	188.0	13.41	182	16396
15_euclidean_1	15	306.7	18.09	224	212	306.7	0.16	210	50	306.7	13.87	210	32781
15_euclidean_2	15	345.9	16.66	224	212	345.9	0.08	210	45	345.9	13.84	210	32781
15_random_asym_1	15	342.9	0.17	224	212	342.9	0.04	210	32	342.9	5.82	210	32781
15_random_sym_1	15	377.5	0.29	224	212	377.5	0.01	210	37	377.5	10.19	210	32781
20_euclidean_1	20	370.1	0.95	399	382	370.1	0.06	380	50	-	-	-	-
20_euclidean_2	20	396.8	9.44	399	382	396.8	0.05	380	56	-	-	-	-
20_random_asym_1	20	324.1	0.43	399	382	324.1	0.02	380	43	-	-	-	-
20_random_sym_1	20	371.1	1.81	399	382	371.1	0.14	380	59	-	-	-	-

TABLE 1: Comparaison complète MTZ vs DFJ Itératif vs DFJ Énumératif

## 3.2 Analyse des Résultats : MTZ vs DFJ

L'objectif de cette sous-section est d'interpréter les écarts de performance observés entre la formulation compacte (MTZ) et la formulation exponentielle (DFJ).

### 3.2.1 Identification des Patterns

L'analyse des contraintes générées met en évidence des différences structurelles majeures :

- **MTZ** : Génère un nombre de contraintes polynomial en  $\mathcal{O}(n^2)$ . Pour  $n = 15$ , cela représente environ 212 contraintes (voir Tableau 1). Ce nombre est fixe et connu à l'avance.
- **DFJ Énumératif** : Génère un nombre exponentiel de contraintes en  $\mathcal{O}(2^n)$ . Pour  $n = 15$ , cela dépasse 32 000 contraintes, ce qui sature la mémoire et ralentit considérablement la construction du modèle avant même le lancement du solveur.
- **DFJ Itératif** : Termine avec un nombre de contraintes extrêmement faible (souvent inférieur à celui de MTZ). Bien que le réservoir théorique de contraintes soit exponentiel, seules celles qui sont violées sont ajoutées dynamiquement. En pratique, il suffit souvent d'ajouter une dizaine de coupes pour prouver l'optimalité.

Concernant le temps de résolution :

- **DFJ Itératif** est systématiquement la méthode la plus rapide (souvent  $< 0.1s$ ).
- **MTZ** est globalement plus lente, avec une dégradation brutale de performance sur certaines topologies (notamment les instances **line**, où le temps peut être multiplié par 10 ou 100 par rapport à DFJ).
- **DFJ Énumératif** devient impraticable dès  $n > 12$  à cause du temps de génération en Python et de la lourdeur du modèle chargé dans le solveur.

### 3.2.2 Influence de la Topologie

La structure des instances joue un rôle crucial dans la performance de MTZ :

- Les instances de type **random** ou **circle** sont résolues assez rapidement par les deux méthodes.
- Les instances de type **line** (villes alignées) constituent le "talon d'Achille" de MTZ. On y observe souvent un *Integrality Gap* (écart entre la solution relaxée et la solution entière) très élevé, parfois supérieur à 40%, alors que DFJ reste imperturbable sur ces instances.

### 3.2.3 Explication Théorique : La Force de la Formulation

La raison pour laquelle la méthode DFJ itérative est plus rapide réside dans la force de sa formulation linéaire.

**Faiblesse de MTZ** La contrainte MTZ s'écrit :

$$u_i - u_j + n \cdot x_{ij} \leq n - 1$$

C'est une contrainte de type "Big-M" (car le coefficient  $n$  devant  $x_{ij}$  est grand). Dans la relaxation continue (où  $x_{ij}$  peut valoir par exemple 0.5), cette contrainte devient très "lâche". Elle autorise des solutions fractionnaires qui sont très loin d'être des tours valides.

**Conséquence :** La borne inférieure (*Lower Bound*) calculée par la relaxation linéaire est mauvaise (trop basse). Le solveur (Branch & Bound) ne peut pas élaguer efficacement l'arbre de recherche et doit explorer des milliers de nœuds pour trouver la solution entière.

**Force de DFJ** Pour DFJ, les contraintes de sous-tours s'écrivent :

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1$$

Ces contraintes définissent très précisément l'enveloppe convexe du polytope du TSP. Même en autorisant des variables continues, il est difficile de "tricher" avec ces contraintes pour former des sous-tours fractionnaires.

**Conséquence :** La valeur de la relaxation continue est souvent égale ou très proche de la solution entière optimale (Gap  $\approx 0\%$ ). Le solveur trouve l'optimum presque immédiatement, souvent à la racine de l'arbre de recherche, sans avoir besoin de brancher excessivement.

### 3.3 MTZ vs DFJ

Caractéristique		Formulation MTZ (Miller–Tucker–Zemlin)	Formulation DFJ (Dantzig–Fulkerson–Johnson)
Complexité d'Implémentation		<b>Faible (Force).</b> Facile à écrire, tient en quelques lignes, pas de boucle externe nécessaire.	<b>Élevée (Faiblesse).</b> Nécessite une boucle itérative, un algorithme de détection de cycles et une gestion dynamique du modèle.
Nombre de Contraintes		<b>Polynomial</b> ( $\mathcal{O}(n^2)$ ). Prévisible et stable.	<b>Exponentiel</b> ( $\mathcal{O}(2^n)$ ). Théoriquement ingérable, mais contourné par l'approche itérative (génération de colonnes/contraintes).
Qualité de la Relaxation		<b>Mauvaise (Faiblesse).</b> Le domaine des solutions relaxées est vaste, entraînant un <i>Integrality Gap</i> important.	<b>Excellente (Force).</b> Le polyèdre est très "serré" autour des solutions entières. Le Gap est souvent nul.
Performance Temps		<b>Moyenne à Lente.</b> Souffre considérablement quand $n$ augmente ou sur des topologies difficiles (lignes).	<b>Rapide (Force).</b> L'approche itérative est actuellement l'une des méthodes les plus efficaces pour résoudre le TSP de manière exacte.
Usage mandé	<b>Recom-</b>	Pour des prototypages rapides, des petits problèmes, ou quand on utilise un solveur basique sans capacité de callbacks.	Pour la recherche de performance pure et la résolution d'instances de taille moyenne à grande.

TABLE 2 – Comparaison synthétique des formulations MTZ et DFJ pour le TSP.

## 4 Analyse de la Relaxation Continue

Le tableau ci-dessous présente les résultats détaillés de la relaxation continue pour l'ensemble des instances testées. Conformément aux consignes, la formulation DFJ Énumératif n'est appliquée qu'aux instances de taille  $n \leq 15$ .

Instance	Formul.	Obj. Entier ( $Z_{IP}$ )	Obj. Relax ( $Z_{LP}$ )	Gap (%)	Tps Relax (s)	Tps Total (s)
10_circle_1	MTZ	298.57	248.58	16.74%	0.005	0.112
	DFJ	298.57	298.57	<b>0.00%</b>	0.035	0.065
10_euclidean_1	MTZ	207.85	200.21	3.68%	0.005	0.073
	DFJ	207.85	207.85	<b>0.00%</b>	0.032	0.064
10_euclidean_2	MTZ	323.92	312.94	3.39%	0.005	0.114
	DFJ	323.92	323.92	<b>0.00%</b>	0.032	0.065
10_line_1	MTZ	156.14	81.49	<b>47.81%</b>	0.005	0.792
	DFJ	156.14	156.14	<b>0.00%</b>	0.034	0.068
10_rand_asym_1	MTZ	191.53	191.53	0.00%	0.005	0.021
	DFJ	191.53	191.53	0.00%	0.032	0.065
10_rand_sym_1	MTZ	280.99	262.29	6.66%	0.005	0.199
	DFJ	280.99	280.99	<b>0.00%</b>	0.033	0.065
12_circle_1	MTZ	265.22	176.08	33.61%	0.006	0.982
	DFJ	265.22	265.22	<b>0.00%</b>	0.191	0.374
12_line_1	MTZ	174.80	111.26	36.35%	0.005	2.933
	DFJ	174.80	174.80	<b>0.00%</b>	0.177	0.365

Instance	Formul.	Obj. Entier ( $Z_{IP}$ )	Obj. Relax ( $Z_{LP}$ )	Gap (%)	Tps Relax (s)	Tps Total (s)
14_circle_1	MTZ	304.48	221.94	27.11%	0.006	1.150
	DFJ	304.48	304.48	<b>0.00%</b>	1.014	2.001
14_line_1	MTZ	188.02	71.59	<b>61.93%</b>	0.007	<b>18.305</b>
	DFJ	188.02	188.02	<b>0.00%</b>	0.951	<b>3.918</b>
15_euclidean_1	MTZ	306.70	228.74	25.42%	0.007	6.458
	DFJ	306.70	306.70	<b>0.00%</b>	2.284	4.489
15_euclidean_2	MTZ	345.91	243.46	29.62%	0.007	5.834
	DFJ	345.91	345.91	<b>0.00%</b>	2.344	4.539
15_rand_asym_1	MTZ	342.94	342.64	0.09%	0.007	<b>0.074</b>
	DFJ	342.94	342.94	<b>0.00%</b>	2.268	4.473
15_rand_sym_1	MTZ	377.50	332.28	11.98%	0.007	0.310
	DFJ	377.50	377.50	<b>0.00%</b>	2.310	4.607
20_euclidean_1	MTZ	370.13	315.75	14.69%	0.010	0.348
20_euclidean_2	MTZ	396.75	336.96	15.07%	0.010	6.523
20_rand_asym_1	MTZ	324.08	319.44	1.43%	0.010	0.470
20_rand_sym_1	MTZ	371.10	356.43	3.95%	0.011	1.890
25_euclidean_1	MTZ	410.26	346.98	15.43%	0.013	3.263
25_euclidean_2	MTZ	418.56	364.41	12.94%	0.014	17.937

TABLE 3: Comparaison des Integrality Gaps et des temps totaux de résolution. Note : Les temps totaux pour DFJ correspondent ici à la méthode Énumérative (seule comparable en relaxation)

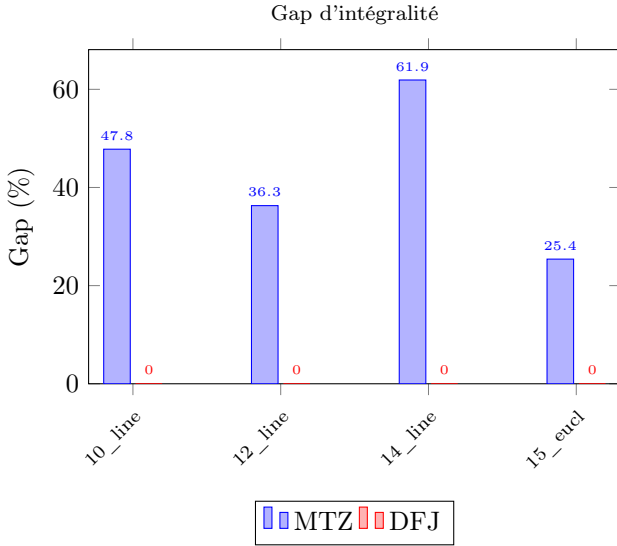


FIGURE 3 – MTZ présente un fort Gap.

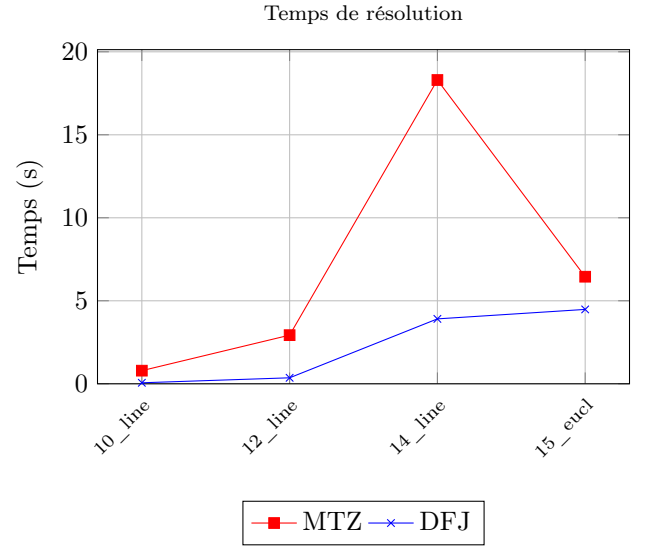


FIGURE 4 – Explosion du temps MTZ.

FIGURE 5 – Comparaison directe : Le Gap élevé de MTZ (gauche) cause l'explosion du temps de résolution (droite), contrairement à la stabilité de DFJ.

Nous observons que le temps de relaxation pour MTZ est plus rapide (complexité en  $\mathcal{O}(n^2)$ ) que pour DFJ. Cependant, cette rapidité de calcul est contrebalancée par un *Integrality Gap* élevé, atteignant notamment **61,93%** sur l'instance **14\_line\_1**. Par conséquent, le solveur est contraint d'explorer plus de solutions, ce qui explique un temps de résolution total bien plus long (18,3 s contre 0,08 s pour DFJ sur cette même instance).

Concernant DFJ, la méthode est beaucoup plus efficace car le gap est souvent de **0%**. Le solveur trouve la solution optimale presque instantanément malgré sa complexité théorique de  $\mathcal{O}(2^n)$ .

## 5 Analyse de la praticabilité : Énumération brute vs Formulation DFJ

Une question fondamentale se pose lors de l'étude du TSP : pourquoi la formulation DFJ est-elle praticable alors qu'elle repose sur un nombre exponentiel de sous-ensembles ( $2^n$ ), tout comme l'énumération brute des cycles ( $\frac{(n-1)!}{2}$ ) ? L'explication de cette différence réside dans trois facteurs clés.

### 5.1 Croissance Factorielle vs Croissance Exponentielle

Bien que les deux approches soient qualifiées d'exponentielles au sens large, leur vitesse de croissance est radicalement différente. L'énumération des cycles suit une progression factorielle. Pour une instance de taille  $n = 20$ , le

nombre de cycles à énumérer est d'environ  $6 \times 10^{16}$ , une valeur qui rend la méthode aveugle totalement impossible à traiter. La formulation DFJ, avec ses  $2^n$  contraintes potentielles, croît moins vite, bien que cela reste insuffisant pour une résolution directe.

## 5.2 L'efficacité de la génération itérative

La praticabilité de la formulation DFJ repose principalement sur le fait qu'elle n'a jamais besoin de manipuler simultanément l'ensemble de ses contraintes théoriques. Contrairement à une approche d'énumération brute, qui impose de considérer explicitement chaque cycle possible, la version itérative de DFJ adopte une stratégie de génération dynamique de contraintes. À chaque itération, seules les contraintes correspondant à des sous-tours effectivement violés par la solution courante du programme linéaire sont ajoutées au modèle, ce qui permet de concentrer l'effort de calcul sur les seules structures problématiques.

Cette approche conduit à une réduction drastique de la taille du modèle traité par le solveur. Les résultats expérimentaux montrent que, pour des instances de taille  $n = 25$ , la convergence vers l'optimum est obtenue avec seulement 72 à 74 contraintes de sous-tours, alors même que le réservoir théorique contient  $2^{25} \approx 33$  millions de sous-ensembles possibles. En pratique, le solveur n'explore donc qu'une fraction infinitésimale de l'espace combinatoire, ce qui explique l'efficacité remarquable de la méthode DFJ malgré sa complexité théorique exponentielle.

## 5.3 La force géométrique de la relaxation (Integrality Gap)

La praticabilité de la formulation DFJ s'explique également par la qualité mathématique de son polyèdre de relaxation. Dans la quasi-totalité des instances testées, la formulation DFJ présente un Integrality Gap nul, ce qui signifie que la solution obtenue par la relaxation linéaire coïncide exactement avec la solution entière optimale. Autrement dit, la relaxation continue est déjà suffisamment restrictive pour éliminer les solutions fractionnaires non valides, ce qui permet au solveur d'identifier immédiatement l'optimum.

À l'inverse, la formulation MTZ se caractérise par une relaxation beaucoup plus « lâche ». Les contraintes de type Big-M autorisent des solutions fractionnaires très éloignées des tournées admissibles, ce qui se traduit par des Integrality Gaps élevés, pouvant atteindre 61,93 % sur certaines instances. Cette faiblesse structurelle contraint le solveur à compenser par une exploration approfondie de l'arbre de recherche via un algorithme de Branch & Bound, augmentant fortement le nombre de nœuds explorés. En conséquence, malgré un nombre initial de contraintes plus faible, la formulation MTZ peut s'avérer paradoxalement plus lente que DFJ sur des instances de taille moyenne à grande.

# 6 Analyse de la supériorité pratique de DFJ sur MTZ

Le paradoxe de ce projet réside dans le fait qu'une formulation possédant un nombre exponentiel de contraintes ( $\mathcal{O}(2^n)$ ) soit plus rapide qu'une formulation compacte polynomiale ( $\mathcal{O}(n^2)$ ). L'explication repose sur la qualité mathématique de la formulation et l'efficacité de la stratégie de résolution.

## 6.1 Efficacité de la génération itérative de contraintes

Contrairement à la formulation MTZ, qui impose l'ensemble de ses contraintes dès la construction initiale du modèle, l'approche DFJ itérative repose sur une stratégie de génération de coupes, souvent qualifiée de lazy constraints. Cette méthode permet d'introduire progressivement les contraintes de sous-tours, uniquement lorsque celles-ci sont nécessaires pour corriger une solution intermédiaire non valide.

Cette stratégie se traduit par une économie significative de contraintes. Pour une instance de taille  $n = 14$ , la formulation MTZ impose systématiquement 184 contraintes, indépendamment de la structure de l'instance. À l'inverse, la formulation DFJ itérative atteint l'optimalité en n'ajoutant qu'entre 34 et 53 contraintes, selon la topologie du graphe. Le solveur ne traite ainsi que les contraintes de sous-tours effectivement violées par la solution courante, tandis que la grande majorité des  $2^n$  contraintes théoriquement possibles ne sont jamais activées. En pratique, cela conduit à un modèle nettement plus léger et plus ciblé que celui issu de la formulation MTZ, ce qui contribue directement aux gains de performance observés.

## 6.2 Force de la formulation et Qualité du polyèdre

La différence de performance observée entre les formulations s'explique principalement par la force de la relaxation linéaire associée à chacune d'elles, notée  $Z_{LP}$ . La formulation MTZ repose sur des contraintes de type Big-M, dans lesquelles le coefficient  $n$  joue le rôle d'une borne artificiellement large. Ce choix rend la relaxation linéaire particulièrement lâche et autorise des solutions fractionnaires très éloignées d'une tournée valide. Cette faiblesse se traduit par un Integrality Gap élevé, pouvant atteindre 61,93 % sur l'instance critique `14_line_1`. Le solveur est alors contraint de compenser cette mauvaise borne inférieure par une exploration intensive de l'arbre de recherche via un algorithme de Branch & Bound.

À l'inverse, les contraintes de la formulation DFJ correspondent à des facettes du polytope du TSP, ce qui confère à la relaxation linéaire une précision bien supérieure. Dans la quasi-totalité des instances testées, cette formulation conduit à un Integrality Gap nul, indiquant que la solution de la relaxation continue coïncide avec la solution entière optimale. Cette propriété a un impact direct sur le processus de résolution : avec un gap nul, le solveur identifie la solution optimale dès la racine de l'arbre de recherche, sans nécessiter de branchements supplémentaires. En pratique, cette différence est particulièrement marquée sur les instances difficiles : pour  $n = 14$  et une topologie de type line, la formulation MTZ nécessite 52,27 s pour converger, contre seulement 0,28 s pour la formulation DFJ itérative.

## 7 Analyse du cas à deux sous-tours

Lorsque l'algorithme détecte une partition exacte en deux sous-tours  $S$  et  $\bar{S}$ , nous avons démontré qu'ajouter une seule contrainte est suffisant.

### 7.1 Preuve de redondance

La contrainte d'élimination de sous-tour pour  $S$  est  $x(S) \leq |S| - 1$ . En utilisant l'identité des degrés ( $2x(S) + x(S, \bar{S}) = 2|S|$ ), cette inégalité est équivalente à la contrainte de coupe :

$$x(S, \bar{S}) \geq 2$$

Puisque la coupure est symétrique ( $x(S, \bar{S}) = x(\bar{S}, S)$ ), imposer la contrainte sur  $S$  force mathématiquement la contrainte sur  $\bar{S}$ . La seconde inégalité est donc redondante.

### 7.2 Vérification expérimentale

Nous avons modifié le code DFJ afin de n'ajouter qu'une seule coupe dans ce cas précis. Le solveur converge néanmoins vers la même solution optimale, ce qui confirme que cette unique contrainte est théoriquement suffisante pour garantir la validité de la solution entière. Sur le plan expérimental, cette simplification n'entraîne toutefois aucun gain de temps notable : par exemple, sur l'instance `15_euclidean`, le temps de résolution reste du même ordre de grandeur (0,15 s contre 0,16 s). Cette observation s'explique par le rôle algorithmique des coupes au-delà de la seule faisabilité entière : même lorsqu'elle est redondante du point de vue de la validité, la seconde contrainte contribue à resserrer la relaxation linéaire. Ainsi, sa suppression réduit bien la taille du modèle, mais affaiblit légèrement le pouvoir de séparation de la relaxation, ce qui tend à annuler le bénéfice attendu en temps de calcul.

## 8 Conclusion

Ce projet a exploré la résolution exacte du Problème du Voyageur de Commerce en confrontant la formulation compacte de Miller-Tucker-Zemlin (MTZ) à l'approche exponentielle de Dantzig-Fulkerson-Johnson (DFJ). Pour départager ces deux modèles, nous avons comparé leurs performances expérimentales en opposant la résolution directe à la stratégie de génération itérative de contraintes. Nos observations démontrent que la simplicité polynomiale de MTZ masque une relaxation linéaire trop faible, tandis que DFJ tire sa robustesse d'un *Integrality Gap* quasi-nul. En définitive, ce travail prouve que la qualité mathématique des contraintes prime sur leur quantité, consacrant la formulation DFJ itérative comme la méthode la plus performante malgré sa complexité théorique.