Recherche locale pour la formation en santé sous contraintes de ressources *

Simon Caillard, Laure Brisoux-Devendeville, Corinne Lucet

Université de Picardie Jules Verne, Laboratoire MIS (EA 4290)
33 rue Saint-Leu, 80039 Amiens Cedex 1, France
{simon.caillard, laure.devendeville, corinne.lucet}@u-picardie.fr

Mots-clés : *recherche opérationnelle, santé, recherche locale, optimisation, planification.*

1 Introduction

SimUSanté, situé à Amiens, est un centre de formation en santé. Utilisé par de nombreux acteurs du domaine, pouvant ainsi s'entraîner ensemble, le centre propose plus de 500 formations différentes. Le problème SimUSanté s'apparente à la fois au problème de Curriculum-Based Courses TimeTabling[4] (CB-CTT), et au problème de Resource-Constrained Project Scheduling[1] (RCPSP). En effet, notre problème intègre simultanément des contraintes de précédence et de disponibilités temporelles des ressources, ainsi que le multi typage de ces dernières. CB-CTT et RCPSP sont NP-Complets[3], le problème de SimUSanté l'est donc aussi.

2 Formalisation

Le problème de SimUSanté est de planifier un ensemble de sessions S sur un horizon T. Chaque session $s \in S$ est composée d'un ensemble d'activités A_s. $A = \bigcup_{s \in S} A_s$ représente l'ensemble des activités. Toute activité $a \in A$ a une durée spécifique $duration_a$ et requiert différents types de ressources en des quantités variables. De plus, il existe un ensemble de ressources R, composé d'employés, de salles et de matériels, où chaque ressource $r \in R$ est associée à un ou plusieurs types de ressources. Une solution Sol au problème, est un ensemble de triplés (a, t_a, R_a) où $a \in A$ est une activité, $t_a \in T$ le créneau de départ de a, et $R_a \subseteq R$ l'ensemble des ressources disponibles affectées à a, pour toute sa durée. Pour une session $s \in S$, $Sol_s \subseteq Sol$ représente l'ensemble des triplés de la solution en rapport à s, avec $Sol_s = \{(a, t_a, R_a) \in Sol | a \in A_s\}$. UA_s représente l'ensemble des activités non planifiées de s. $UA = \bigcup_{s \in S} UA_s$ est alors l'ensemble des activités non planifiées. L'évaluation de Sol est donnée par $Makespan(Sol) = \sum_{s \in S}(\max\{t_a + duration_a, a \in Sol_s\} - \min\{t_a, a \in Sol_s\}) + |UA| \times \alpha$. L'objectif est alors de trouver une solution qui minimise $Makespan$.

3 SimuLS, un algorithme de recherche locale

SimuLS est un algorithme de recherche locale qui explore l'espace des solutions en appliquant des opérateurs de voisinage à partir d'une solution fournie par l'algorithme glouton *SimuG*[2]. Durant un nombre préfixé d'itérations, *SimuLS* applique trois types d'opérateurs : un opérateur de complétion de la solution *saturator*, trois opérateurs d'exploration du voisinage *intra*, *extra* et *extra+*, et un opérateur de diversification *destructor*.

Un mouvement est défini par un couple $< (a, t_a, R_a)\ ;\ \Upsilon >$. (a, t_a, R_a) représente un triplé qui sera ajouté à la solution courante, avec $a \in UA$, une activité non planifiée, $t_a \in T$, un créneau horaire à partir duquel a peut commencer, et R_a l'ensemble des ressources affectées à a, qui correspond à ses besoins. Pour planifier a, un ensemble Υ de triplés de la solution doit être supprimé, avec $\Upsilon = \{(b_1, t_{b_1}, R_{b_1}), \ldots, (b_n, t_{b_n}, R_{b_n})\}$, $n \in \{1, \ldots, |Sol|\}$. A l'exception

*Ce projet est cofinancé par la région Hauts-de-France et le centre de simulation en santé SimUSanté

de *destructor*, tout opérateur construit un ensemble de mouvement M :$\{< (a, t_a^1, R_a); \Upsilon > , \ldots, < (a, t_a^k, R_a); \Upsilon >\}$, sélectionne un mouvement $m \in M$, et l'applique. Le choix de m dépend de critères comme le mk_s des sessions impactées, $Makespan(Sol)$, le nombre d'activités déplanifiées, etc.

L'opérateur de complétion *saturator* permet, pour une session $s \in S$ de programmer une activité $a \in UA_s$ sans déprogrammer d'activités de la solution ($\forall m \in M$, $\Upsilon = \emptyset$). Les opérateurs d'exploration déprogramment toujours une ou plusieurs activités ($\Upsilon \neq \emptyset$). Pour une session $s \in S$, *intra* déprogramme un ou plusieurs triplés de Sol_s ($\Upsilon \subseteq Sol_s$) afin de programmer une activité $a \in UA_s$. L'opérateur *extra* sélectionne aléatoirement une session $s' \neq s$ et déprogramme un ou plusieurs triplés de $Sol_{s'}$ ($\Upsilon \subseteq Sol_{s'}$) pour programmer une activité $a \in UA_s$. L'opérateur $extra^+$ déprogramme un ou plusieurs triplés de Sol ($\Upsilon \subseteq Sol$), afin de programmer une activité $a \in UA_s$. Enfin, l'opérateur *destructor* construit et applique un ensemble de mouvements M :$\{< \emptyset \ ; \ \{(a_1, t_{a_1}, R_{a_1}\} >, \ldots, < \emptyset \ ; \ \{(a_k, t_{a_k}, R_{a_k}\} >$ où $\forall i \in [1; k]$, $(a_i, t_{a_i}, R_{a_i}) \in Sol$ représente un triplé qui sera supprimé de Sol.

Afin de déterminer quel opérateur choisir parmi *intra*, *extra*, $extra^+$, *SimuLS* utilise deux compteurs spécifiques c^s_{extra} et $c^a_{extra^+}$. Le premier, c^s_{extra}, détermine le nombre de fois où *intra* a été consécutivement appliqué à la session s. Le second compte combien de fois consécutives l'activité a est restée non planifiée. Par défaut, l'opérateur *intra* est appliqué, sauf lorsqu'un de ces compteurs atteint une limite préfixée, l'opérateur correspondant est alors utilisé.

4 Resultats

SimuLS a été testé sur différentes instances, issues du problème CB-CTT [1], et adaptées au problème de SimUSanté. *SimuLS* est implémenté en Java, sur un processeur intel i7 7500U. Les résultats obtenus par *SimuLS* furent comparés à ceux obtenus par le solveur CPLEX ainsi qu'aux résultats de l'algorithme glouton *SimuG* sur les 48 instances générées. *SimuG* obtient des résultats en moins d'une seconde, et *SimuLS* en moins d'une minute. Néanmoins une fois sur deux, *SimuG* ne place pas toutes les activités, et l'écart avec les solutions optimales est compris entre 3 et 246%. *SimuLS* améliore toujours les résultats de *SimuG*, et l'écart avec l'optimalité se situe entre 0 et 18%. De plus, l'ensemble des activités est toujours planifié.

5 Conclusion

Nous avons présenté un algorithme de recherche locale pour résoudre un problème de planification dans le domaine de la formation en santé. Il est basé sur 5 opérateurs de voisinage, ou quatre d'entre eux permettre de planifier des activités et un de détruire partiellement la solution afin d'échapper à un minimum local. *SimuLS* a été testé sur des instances du CB-CTT, adaptées au problème de SimUSanté. Les résultats de *SimuLS* ont été comparés aux solutions optimales obtenues sous CPLEX, et contrairement à *SimuG*, toutes les activités sont placées. Ce travail est une première étape dans la construction d'une méthode permettant de traiter plus largement le problème de planification que rencontre le centre de formation SimUSanté.

Références

[1] P. Brucker and S. Knust. Resource-Constrained Project Scheduling and Timetabling. In *Practice and Theory of Automated Timetabling III*, pages 277–293. Springer, 2001.

[2] S. Caillard, L. Devendeville, and C. Lucet. A Planning Problem with Resource Constraints in Health Simulation Center. In *Optimization of Complex Systems*. Springer, 2020.

[3] T. Cooper and J. Kingston. The Complexity of Timetable Construction Problems. In *Practice and Theory of Automated Timetabling*, pages 281, 295. Springer, 1996.

[4] L. Di Gaspero, B. McCollum, and A. Schaerf. *Curriculum-based CTT - Technical Report*. The Second Int. Timetabling Competition (ITC-2007).

1. Les instances SimUSanté sont disponibles à l'adresse : `https://mis.u-picardie.fr/Benchmarks-GOC`