# Influential Article Review - Profit-driven Scheme For Limited-Resource Projects With Adjustable Capacity Limits

**Kofi Hinton**

**Shola Potter**

**Connah Kumar**

*This paper examines operations. We present insights from a highly influential paper. Here are the highlights from this paper: We consider a novel generalization of the resource-constrained project scheduling problem (RCPSP). Unlike many established approaches for the RCPSP that aim to minimize the makespan of the project for given static capacity constraints, we consider the important real-life aspect that capacity constraints can often be systematically modified by temporarily assigning costly additional production resources or using overtime. We, furthermore, assume that the revenue of the project decreases as its makespan increases and try to find a schedule with a profit-maximizing makespan. Like the RCPSP, the problem is NP-hard, but unlike the RCPSP, it turns out that an optimal schedule does not have to be among the set of so-called active schedules. Scheduling such a project is a formidable task, both from a practical and a theoretical perspective. We develop, describe, and evaluate alternative solution encodings and schedule decoding mechanisms to solve this problem within a genetic algorithm framework and we compare the solutions obtained to both optimal reference values and the results of a commercial local search solver called LocalSolver. For our overseas readers, we then present the insights from this paper in Spanish, French, Portuguese, and German.*

*Keywords: Project scheduling, Encodings, Heuristics, Local-search, Genetic algorithm, RCPSP, Overtime*

## SUMMARY

- For large projects with 120 activities, the exact method is not able to produce any reasonably useful results within a 2 min time limit. The heuristic methods still yield fairly good solutions with small gaps towards the best known solutions. The dominance of LocalSolver models using indirect solution encodings over the problem-specific genetic algorithm counterparts is now broken and flipped. The specific genetic algorithms clearly beat all other procedures considered when solving larger problem instances. Therefore, the low-effort solution method of using a standard solver is not advisable for large instances. In addition, it seems that the ðkjz^rtÞ representation is not very well suited for solving large problem instances in conjunction with LocalSolver. Although algorithmically the problem-specific approach is very likely to be superior, the generalized local search implementation is a commercial software product with years of development and effort by a team of programmers, whereas the genetic algorithm was implemented in a shorter time from

scratch. This might explain why a black-box heuristic solver is able to outperform a problem-specific genetic algorithm for small problem instances. For large instances, the algorithmic advantages from problem knowledge in the genetic algorithms seem to dominate any implementation issues in comparison to a commercially developed and optimized general-purpose software.

- In this paper, we presented an extension of the RCPSP with overtime cost and a revenue function monotonically decreasing with project duration. We formalized the scheduling problem as a mixed-integer linear program and designed encodings as preparation step for the development of efficient solution procedures. For future research, it is promising to use modified operators of the genetic algorithm to achieve better results, for example, the peak crossover operator proposed by Valls et al. . This operator considers the fitness of the individuals in the crossover.
- In addition, the activity list as the core of all representations evaluated may be exchanged with another widespread encoding for inducing activity priorities, the socalled standardized random key representation, Debels et al. .
- However, it is expected that the general solution behavior remains the same even with such improvements. Therefore, it would be even more interesting to use entirely different solution procedures or representations.

## HIGHLY INFLUENTIAL ARTICLE

We used the following article as a basis of our evaluation:

Schnabel, A., Kellenbrink, C., & Helber, S. (2018). Profit-oriented scheduling of resource-constrained projects with flexible capacity constraints. Business Research, 11(2), 329–356.

This is the link to the publisher's website:
https://link.springer.com/article/10.1007/s40685-018-0063-5

## INTRODUCTION

Many models and procedures for resource-constrained project scheduling problems (RCPSPs) assume that the capacities of the renewable resources that are required to perform the project's activities are exogenously given and that the objective is to find a (feasible) schedule with a minimal project makespan or duration. In reality, the renewable resources like human labor or machinery are often temporarily assigned to a project and decisions on additional resources or overtime are made to achieve a short project duration. A short project duration may be economically attractive for different reasons. Consider, for example, software development projects or construction projects for factories. In such cases, a shorter project duration may permit an earlier market entry. This desire to achieve a short project duration can, e.g., lead to contractual penalty clauses or other incentive schemes that relate actual payments to project durations. In such cases, the revenue from a project typically decreases as its duration increases. This immediately leads to the question how to use overtime and how to schedule such projects with flexible capacity constraints and makespan-dependent revenues in the most profitable way.

The remainder of this paper is organized as follows: in Sect. 2, we describe the assumptions of the resource-constrained project scheduling problem with makespan-specific revenues and option of overcapacity (RCPSP-ROC), give real-world examples, demonstrate basic problem and solution properties, and provide an overview of the related literature. In Sect. 3, we develop a formal mathematical decision model for the RCPSP-ROC and discuss properties of solutions that guide the development of solution procedures. The design rationales and detailed descriptions of different solution encodings for this problem are given in Sect. 4.2. On this basis, we propose various genetic algorithms and local search procedures in Sects. 4.3 and 4.4. Section 5 is devoted to the test design and the results from a numerical study to evaluate

the different proposed methods to solve the RCPSP-ROC. Sect. 6 concludes this paper by giving a short summary of the results and suggestions for future research.

## CONCLUSION

In this paper, we presented an extension of the RCPSP with overtime cost and a revenue function monotonically decreasing with project duration. We formalized the scheduling problem as a mixed-integer linear program and designed encodings as preparation steps for the development of efficient solution procedures. We then developed a genetic algorithm for the problem, computed and interpreted results for a problem library based on a widely used RCPSP test set. We further investigated the use of a general local search heuristic, thus offering numerical results for both problem specific as well as generic black-box heuristic solution methods. The results are very promising. For larger projects with many activities, heuristic problem-specific solution methods beat generic black-box inexact solvers. For small size projects, using a heuristic black-box method worked best.

For future research, it is promising to use modified operators of the genetic algorithm to achieve better results, for example, the peak crossover operator proposed by Valls et al. (2008). This operator considers the fitness of the individuals in the crossover.

In addition, the activity list as the core of all representations evaluated may be exchanged with another widespread encoding for inducing activity priorities, the so-called standardized random key representation, Debels et al. (2006).

However, it is expected that the general solution behavior remains the same even with such improvements. Therefore, it would be even more interesting to use entirely different solution procedures or representations. A suitable and ideally more compact solution encoding may speed up the solution process by removing more redundant points in the solution space. One idea is to evaluate a representation based on quasi-stable schedules known for resource-leveling problems with a heuristically defined makespan. Again, the goal is to explore the smallest possible set guaranteed to contain an optimal solution. However, to this end and also to develop an exact algorithm, it would be extremely helpful to identify and formalize properties of optimal solutions.

## APPENDIX

**FIGURE 1**
**ACTIVITY ON NODE GRAPH**

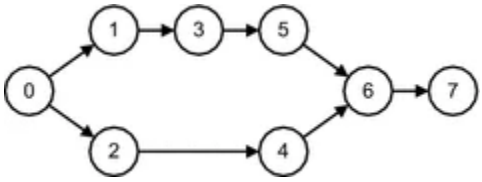


**FIGURE 2**

**OPTIMAL SCHEDULE WITHOUT OVERTIME USAGE FOR CUSTOMER A**

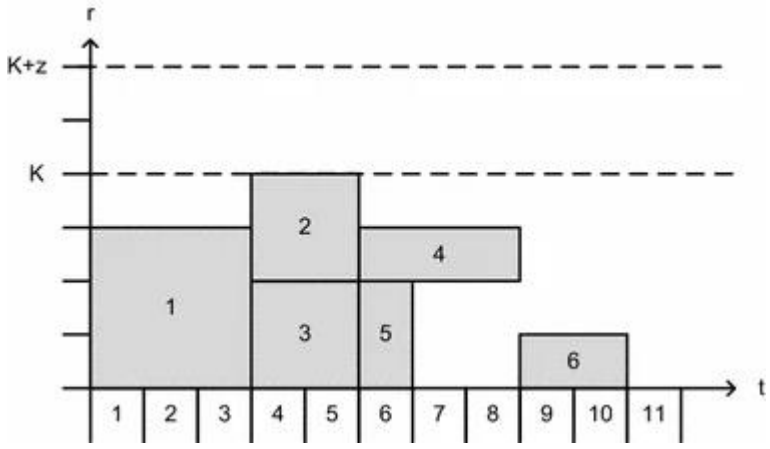


**FIGURE 3**
**SCHEDULE WITH MAXIMUM AMOUNT OF OVERTIME UTILIZED**

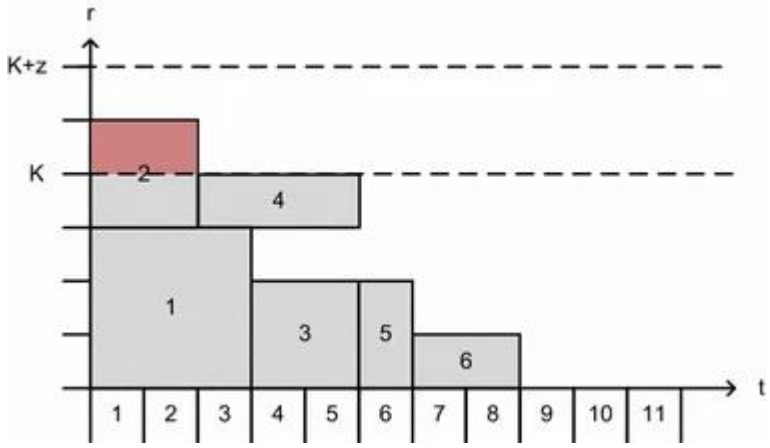


**FIGURE 4**
**SCHEDULE WITH SOME OVERTIME USAGE**

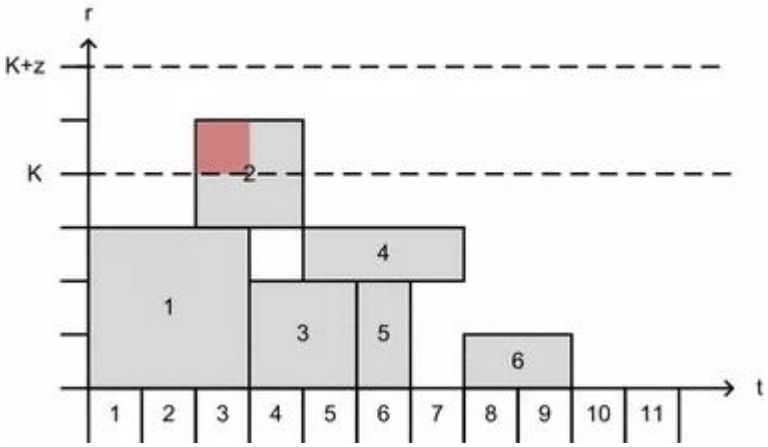


**FIGURE 5**
**RELATIONSHIP BETWEEN MAKESPAN, OVERTIME COST, REVENUE AND PROFIT**

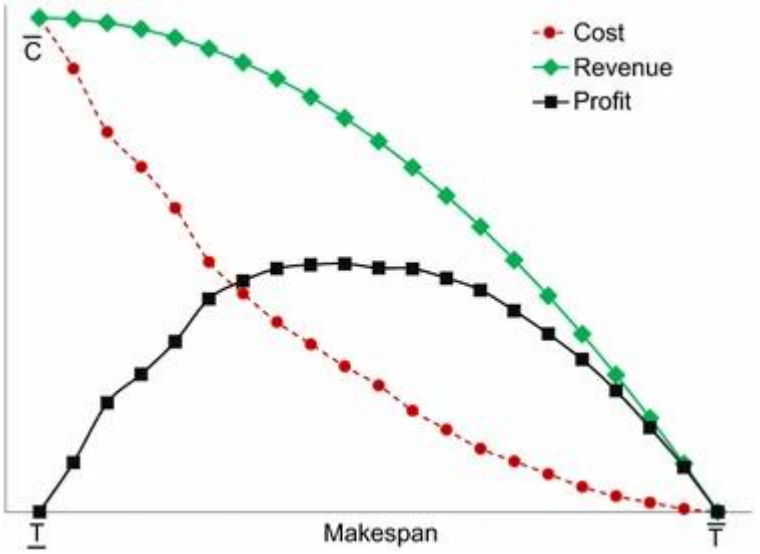


**FIGURE 6**
**EXAMPLE OF OPTIMAL SCHEDULE BEING OUTSIDE OF QSS**

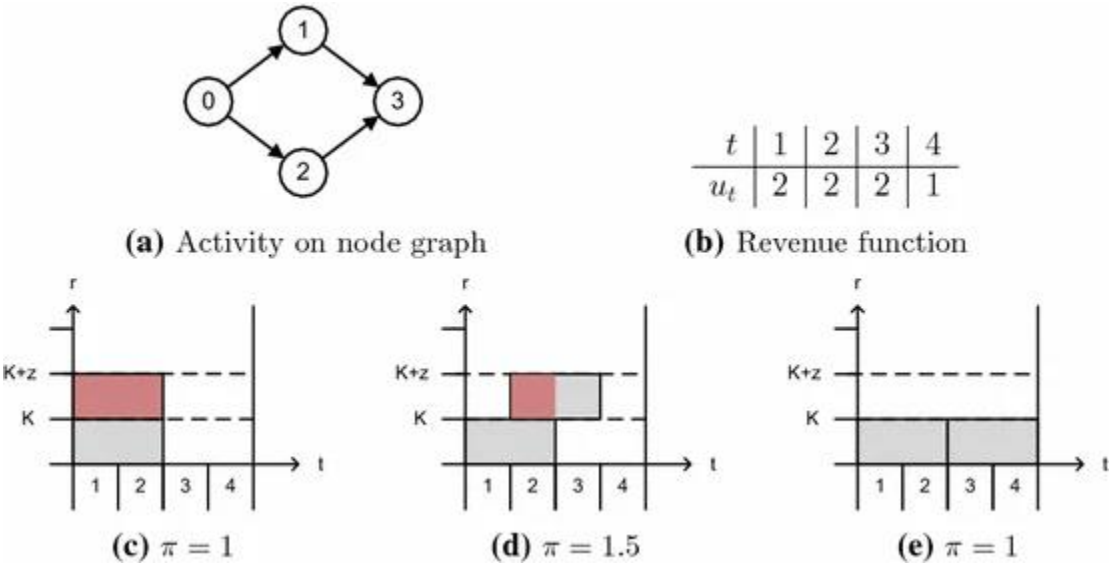


**TABLE 1**
**ACTIVITY DURATIONS AND CAPACITY REQUIREMENTS**

| Activity *j* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Duration djdj | 0 | 3 | 2 | 2 | 3 | 1 | 2 | 0 |
| Capacity requirement kj,1kj,1 | 0 | 3 | 2 | 2 | 1 | 2 | 1 | 0 |

**TABLE 2**
**MAKESPAN-DEPENDENT WILLINGNESS TO PAY (REVENUE) OF DIFFERENT CUSTOMERS**

| Makespan | < 8 | 8 | 9 | 10 | 11 | > 11 |
|---|---|---|---|---|---|---|
| Customer A | 10 | 10 | 10 | 10 | 10 | 0 |
| Customer B | 60 | 45 | 30 | 15 | 0 | 0 |
| Customer C | 20 | 20 | 15 | 0 | 0 | 0 |

**TABLE 3**
**NOTATION OF THE RCPSP-ROC**

| Indices and (ordered) sets | |
|---|---|
| j∈Jj∈J | Activities J={0,1,…,J,J+1}J={0,1,…,J,J+1} |
| t,τ∈Tt,τ∈T | Periods T={0,1,…,T}T={0,1,…,T} |
| r∈Rr∈R | Renewable resources R={1,…,R}R={1,…,R} |

| | |
|---|---|
| Pj⊆JPj⊆J | Set of immediate predecessors of activity *j* |
| Parameters | |
| djdj | Duration of activity *j* |
| EFTjEFTj | Earliest finishing time of activity *j* |
| LFTjLFTj | Latest finishing time of activity *j* |
| kjrkjr | Required units of resource *r* while executing activity *j* |
| KrKr | Capacity of resource *r* |
| z¯¯¯rz¯r | Overtime limit of resource *r* |
| κrκr | Per-unit cost for overtime of resource *r* |
| utut | Revenue for project completion at the end of period *t* |
| Decision variables | |
| xjtxjt | ={1,0,ifactivityjisfinished at theendofperiodtotherwise={1,ifactivityjisfinished at theendofperiodt0,otherwise |
| zrtzrt | Amount of overtime of resource *r* used in period *t* |

**TABLE 4**
**NUMERICAL RESULTS OVER TIME FOR SMALL PROJECTS WITH 30 NON-DUMMY ACTIVITIES**

| Time | Gurobi | Genetic Algorithm | | | LocalSolver | | |
|---|---|---|---|---|---|---|---|
| | | $(\lambda\|\beta)$ | $(\lambda\|\hat{z}_r)$ | $(\lambda\|\hat{z}_{rt})$ | $(\lambda\|\beta)$ | $(\lambda\|\hat{z}_r)$ | $(\lambda\|\hat{z}_{rt})$ |
| 0 | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| 0.01 | 100.00% | 67.60% | 64.66% | 69.72% | 82.50% | 88.46% | 82.51% |
| 0.02 | 100.00% | 4.41% | 4.41% | 3.59% | 36.14% | 50.68% | 45.57% |
| 0.03 | 97.41% | 1.86% | 1.84% | 1.30% | 5.30% | 5.41% | 20.82% |
| 0.04 | 95.02% | 1.26% | 1.22% | 0.95% | 3.29% | 2.20% | 16.23% |
| 0.05 | 90.44% | 1.04% | 1.06% | 0.74% | 2.04% | 1.72% | 13.75% |
| 0.06 | 79.52% | 0.82% | 0.90% | 0.58% | 1.45% | 1.46% | 12.09% |
| 0.07 | 70.58% | 0.69% | 0.77% | 0.49% | 1.25% | 1.23% | 10.59% |
| 0.08 | 64.21% | 0.63% | 0.65% | 0.42% | 1.06% | 1.15% | 9.40% |
| 0.09 | 57.38% | 0.57% | 0.53% | 0.36% | 1.00% | 1.04% | 8.53% |
| 0.1 | 50.64% | 0.53% | 0.50% | 0.32% | 0.91% | 0.93% | 6.92% |
| 0.11 | 46.77% | 0.52% | 0.47% | 0.29% | 0.86% | 0.89% | 6.01% |
| 0.12 | 43.71% | 0.47% | 0.43% | 0.28% | 0.79% | 0.83% | 5.38% |
| 0.32 | 21.20% | 0.19% | 0.35% | 0.14% | 0.46% | 0.47% | 1.53% |
| 0.54 | 15.59% | 0.16% | 0.33% | 0.12% | 0.30% | 0.38% | 0.84% |
| 1 | 9.62% | 0.13% | 0.32% | 0.11% | 0.19% | 0.31% | 0.53% |
| 2 | 6.58% | 0.13% | 0.31% | 0.09% | 0.14% | 0.25% | 0.28% |
| 3 | 5.26% | 0.12% | 0.31% | 0.09% | 0.10% | 0.23% | 0.21% |
| 4 | 4.55% | 0.12% | 0.31% | 0.09% | 0.09% | 0.20% | 0.18% |
| 5 | 3.96% | 0.11% | 0.31% | 0.09% | 0.09% | 0.17% | 0.12% |
| 28 | 1.51% | 0.10% | 0.28% | 0.08% | 0.02% | 0.12% | 0.03% |
| 29 | 1.51% | 0.10% | 0.28% | 0.08% | 0.02% | 0.12% | 0.03% |
| 30 | 1.46% | 0.10% | 0.28% | 0.08% | 0.02% | 0.12% | 0.03% |
| ØGap | 1.46% | 0.10% | 0.28% | 0.08% | 0.02% | 0.12% | 0.03% |
| %Optimal | 38.80% | 39.49% | 36.92% | 41.03% | 43.93% | 42.56% | 43.59% |
| Max Gap | 100.00% | 3.07% | 20.00% | 2.20% | 1.00% | 20.00% | 2.98% |

**TABLE 5**
**NUMERICAL RESULTS FOR SMALL PROJECTS WITH 30 NON-DUMMY ACTIVITIES**

| #Schedules | Genetic algorithm | | | Local Solver | | |
|---|---|---|---|---|---|---|
| | (λ\|β) | (λ\|z^r) | (λ\|z^rt) | (λ\|β) | (λ\|z^r) | (λ\|z^rt) |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1000 | 1.60% | 1.52% | 1.31% | 3.00% | 2.57% | 15.31% |
| 5000 | 0.50% | 0.57% | 0.32% | 0.67% | 0.76% | 4.51% |
| 50,000 | 0.14% | 0.33% | 0.12% | 0.15% | 0.29% | 0.33% |

**TABLE 6**
**NUMERICAL RESULTS OVER TIME FOR LARGE PROJECTS WITH 120 NON-DUMMY ACTIVITIES**

| Time | Gurobi | Genetic Algorithm | | | LocalSolver | | |
|---|---|---|---|---|---|---|---|
| | | $(\lambda\|\beta)$ | $(\lambda\|\hat{z}_r)$ | $(\lambda\|\hat{z}_{rt})$ | $(\lambda\|\beta)$ | $(\lambda\|\hat{z}_r)$ | $(\lambda\|\hat{z}_{rt})$ |
| 0 | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% |
| 0.01 | 100.00% | 69.93% | 68.84% | 69.14% | 100.00% | 100.00% | 100.00% |
| 0.02 | 100.00% | 45.43% | 45.72% | 45.94% | 73.63% | 71.17% | 71.95% |
| 0.03 | 100.00% | 36.51% | 36.50% | 36.29% | 53.25% | 46.56% | 56.16% |
| 0.04 | 100.00% | 31.25% | 32.28% | 31.66% | 48.22% | 39.92% | 51.52% |
| 0.34 | 97.58% | 5.32% | 4.60% | 4.61% | 29.56% | 9.27% | 38.36% |
| 0.52 | 93.87% | 4.39% | 3.81% | 3.84% | 25.66% | 6.68% | 36.46% |
| 0.78 | 87.98% | 3.74% | 3.16% | 3.22% | 21.98% | 5.40% | 34.85% |
| 1 | 85.65% | 3.24% | 2.69% | 2.72% | 19.60% | 4.58% | 33.99% |
| 2 | 81.76% | 2.46% | 1.92% | 1.90% | 13.76% | 3.56% | 31.22% |
| 3 | 77.53% | 2.11% | 1.57% | 1.57% | 11.21% | 3.25% | 28.77% |
| 4 | 76.09% | 1.88% | 1.35% | 1.35% | 9.50% | 3.00% | 26.76% |
| 5 | 74.03% | 1.73% | 1.20% | 1.20% | 8.51% | 2.80% | 25.00% |
| 6 | 71.93% | 1.61% | 1.09% | 1.08% | 7.77% | 2.69% | 23.55% |
| 7 | 71.10% | 1.53% | 1.02% | 1.00% | 7.27% | 2.61% | 22.22% |
| 8 | 70.17% | 1.45% | 0.96% | 0.92% | 6.84% | 2.54% | 21.13% |
| 9 | 69.33% | 1.38% | 0.92% | 0.87% | 6.58% | 2.50% | 20.10% |
| 10 | 68.33% | 1.33% | 0.87% | 0.82% | 6.32% | 2.46% | 19.11% |
| 60 | 52.39% | 0.65% | 0.63% | 0.38% | 3.81% | 1.73% | 9.29% |
| 80 | 50.39% | 0.57% | 0.61% | 0.35% | 3.43% | 1.64% | 8.71% |
| 100 | 48.67% | 0.50% | 0.60% | 0.33% | 3.26% | 1.58% | 8.23% |
| 120 | 47.34% | 0.45% | 0.59% | 0.32% | 3.09% | 1.55% | 7.91% |
| ØGap | 47.34% | 0.45% | 0.59% | 0.32% | 3.09% | 1.55% | 7.91% |
| %Best | 29.57% | 46.84% | 35.04% | 52.31% | 27.35% | 30.94% | 25.13% |
| Max Gap | 100.00% | 2.94% | 4.17% | 3.60% | 46.47% | 7.39% | 38.91% |

**TABLE 7**
**NUMERICAL RESULTS FOR LARGE PROJECTS WITH 120 NON-DUMMY ACTIVITIES**

| #Schedules | Genetic Algorithm | | | LocalSolver | | |
|---|---|---|---|---|---|---|
| | (λ\|β) | (λ\|z^r) | (λ\|z^rt) | (λ\|β) | (λ\|z^r) | (λ\|z^rt) |
| 1000 | 4.97% | 4.17% | 4.19% | 31.22% | 11.73% | 39.26% |
| 5000 | 3.40% | 2.73% | 2.82% | 16.33% | 3.83% | 32.70% |
| 50,000 | 1.39% | 0.86% | 0.85% | 5.29% | 2.00% | 14.22% |

**REFERENCES**


Artigues, C., O. Koné, P. Lopez, and M. Mongeau. 2015. Mixed-integer linear programming formulations. In Handbook on Project Management and Scheduling, vol. 1, ed. C. Schwindt, and J. Zimmermann, 17–41. International Handbooks on Information Systems. Cham Heidelberg New York Dordrecht London: Springer.

Ballestin, F., and R. Blanco. 2015. Theoretical and practical fundamentals. In Handbook on Project Management and Scheduling, vol. 1, ed. C. Schwindt, and J. Zimmermann, 411–427. International Handbooks on Information Systems. Cham Heidelberg New York Dordrecht London: Springer.

Benoist, T., B. Estellon, F. Gardi, R. Megel, and K. Nouioua. 2011. Localsolver 1.x: a black-box local-search solver for 0–1 programming. 4OR 9: 299.

Blazewicz, J., J.K. Lenstra, and A.R. Kan. 1983. Scheduling subject to resource constraints: classification and complexity. Discrete Applied Mathematics 5: 11–24.

Brucker, P., A. Drexl, R. Möhring, K. Neumann, and E. Pesch. 1999. Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research 112: 3–41.

Brucker, P., and S. Knust. 2012. Complex Scheduling. Berlin Heidelberg: Springer.

Debels, D., B. De Reyck, R. Leus, and M. Vanhoucke. 2006. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. European Journal of Operational Research 169: 638–653.

Deckro, R., and J. Hebert. 1989. Resource constrained project crashing. Omega 17: 69–79.

Demeulemeester, E.L., and W.S. Herroelen. 2006. International Series in Operations Research and Management Science. Project scheduling: a research handbook, vol. 49. New York: Springer.

Drexl, A., and A. Kimms. 2001. Optimization guided lower and upper bounds for the resource investment problem. The Journal of the Operational Research Society 52: 340–351.

Easa, S.M. 1989. Resource leveling in construction by optimization. Journal of Construction Engineering and Management 115: 302–316.

Hartmann, S. 1998. A competitive genetic algorithm for resource-constrained project scheduling. Naval Research Logistics (NRL) 45: 733–750.

Hartmann, S. 1999. Lecture Notes in Economics and Mathematical Systems. Project Scheduling under Limited Resources: Models, Methods, and Applications, vol. 478. Berlin Heidelberg: Springer.

Hartmann, S. 2012. Project scheduling with resource capacities and requests varying with time: a case study. Flexible Services and Manufacturing Journal 25: 74–93.

Hartmann, S. 2015. Time-varying resource requirements and capacities. In Handbook on Project Management and Scheduling, vol. 1, ed. C. Schwindt, and J. Zimmermann, 163–176. International Handbooks on Information Systems. Cham Heidelberg New York Dordrecht London: Springer.

Hartmann, S., and D. Briskorn. 2010. A survey of variants and extensions of the resource-constrained project scheduling problem. European Journal of Operational Research 207: 1–14.
Herroelen, W., E. Demeulemeester, and B. De Reyck. 1998. Resource-constrained project scheduling: a survey of recent developments. Computers & Operations Research 25: 279–302.
Hindelang, T.J., and J.F. Muth. 1979. A dynamic programming algorithm for decision CPM networks. Operations Research 27: 225–241.
Holland, J.H. 1975. Adaption in Natural and Artificial Systems: An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence. Detroit: The University of Michigan Press.
Kellenbrink, C., and S. Helber. 2015. Scheduling resource-constrained projects with a flexible project structure. European Journal of Operational Research 246: 379–391.
Kellenbrink, C., and S. Helber. 2016. Quality-and profit-oriented scheduling of resource-constrained projects with flexible project structure via a genetic algorithm. European Journal of Industrial Engineering 10: 574–595.
Klein, R. 2000. Project scheduling with time-varying resource constraints. International Journal of Production Research 38: 3937–3952.
Kolisch, R. 1995. Project scheduling under resource constraints: efficient heuristics for several problem classes. Heidelberg: Physica.
Kolisch, R., and A. Drexl. 1996. Adaptive search for solving hard project scheduling problems. Naval Research Logistics (NRL) 43: 23–40.
Kolisch, R., and S. Hartmann. 1999. Heuristic algorithms for the resource-constrained project scheduling problem: classification and computational analysis. In Project Scheduling, ed. J. Węglarz, 147–178. International Series in Operations Research & Management Science. Boston, MA, USA: Springer.
Kolisch, R., and S. Hartmann. 2006. Experimental investigation of heuristics for resource-constrained project scheduling: An update. European Journal of Operational Research 174: 23–37.
Kolisch, R., and R. Padman. 2001. An integrated survey of deterministic project scheduling. Omega 29: 249–272.
Kolisch, R., and A. Sprecher. 1996. PSPLIB—A project scheduling problem library. European Journal of Operational Research 96: 205–216.
Li, K., and R. Willis. 1992. An iterative scheduling technique for resource-constrained project scheduling. European Journal of Operational Research 56: 370–379.
Neumann, K., H. Nübel, and C. Schwindt. 2000. Active and stable project scheduling. Mathematical Methods of Operations Research 52: 441–465.
Neumann, K., C. Schwindt, and J. Zimmermann. 2003. Project scheduling with time windows and scarce resources. Berlin Heidelberg: Springer.
Neumann, K., and J. Zimmermann. 1999. Resource levelling for projects with schedule-dependent time windows. European Journal of Operational Research 117: 591–605.
Özdamar, L., and G. Ulusoy. 1995. A survey on the resource-constrained project scheduling problem. IIE Transactions 27: 574–586.
Pritsker, A.A.B., L.J. Waiters, and P.M. Wolfe. 1969. Multiproject scheduling with limited resources: A zero-one programming approach. Management Science 16: 93–108.
Schwindt, C. 2005. Resource allocation in project management. Berlin Heidelberg: Springer.
Tormos, P., and A. Lova. 2001. A competitive heuristic solution technique for resource-constrained project scheduling. Annals of Operations Research 102: 65–81.
Valls, V., F. Ballestin, and S. Quintanilla. 2005. Justification and RCPSP: A technique that pays. European Journal of Operational Research 165: 375–386.
Valls, V., F. Ballestin, and S. Quintanilla. 2008. A hybrid genetic algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research 185: 495–508.

**TRANSLATED VERSION: SPANISH**

Below is a rough translation of the insights presented above. This was done to give a general understanding of the ideas presented in the paper. Please excuse any grammatical mistakes and do not hold the original authors responsible for these mistakes.

**VERSION TRADUCIDA: ESPAÑOL**

A continuación se muestra una traducción aproximada de las ideas presentadas anteriormente. Esto se hizo para dar una comprensión general de las ideas presentadas en el documento. Por favor, disculpe cualquier error gramatical y no responsabilite a los autores originales de estos errores.

**INTRODUCCIÓN**

Muchos modelos y procedimientos para problemas de programación de proyectos (RCPSP) con recursos limitados suponen que las capacidades de los recursos renovables necesarios para realizar las actividades del proyecto se dan exógenamente y que el objetivo es encontrar una programación (factible) con un principio o duración mínima del proyecto. En realidad, los recursos renovables como la mano de obra humana o la maquinaria a menudo se asignan temporalmente a un proyecto y se toman decisiones sobre recursos adicionales o horas extras para lograr una duración corta del proyecto. Una duración corta del proyecto puede ser económicamente atractiva por diferentes razones. Consideremos, por ejemplo, proyectos de desarrollo de software o proyectos de construcción para fábricas. En tales casos, una duración más corta del proyecto puede permitir una entrada en el mercado anterior. Este deseo de lograr una duración corta del proyecto puede, por ejemplo, dar lugar a cláusulas contractuales de penalización u otros esquemas de incentivos que relacionen los pagos reales con las duraciones del proyecto. En tales casos, los ingresos de un proyecto normalmente disminuyen a medida que aumenta su duración. Esto lleva inmediatamente a la pregunta de cómo utilizar las horas extras y cómo programar estos proyectos con restricciones de capacidad flexibles y los ingresos dependientes de makespande la manera más rentable.

El resto de este documento se organiza de la siguiente manera: en la sección 2, describimos los supuestos del problema de programación de proyectos con recursos limitados coningresos específicos y la opción de sobrecapacidad (RCPSP-ROC), damos ejemplos del mundo real, demostramos propiedades básicas de problemas y soluciones, y proporcionamos una visión general de la literatura relacionada. En la Secta 3, desarrollamos un modelo de decisión matemática formal para el RCPSP-ROC y discutimos las propiedades de las soluciones que guían el desarrollo de procedimientos de solución. Las razones de diseño y las descripciones detalladas de las diferentes codificaciones de soluciones para este problema se indican en la sección 4.2. Sobre esta base, proponemos varios algoritmos genéticos y procedimientos de búsqueda local en Sects. 4.3 y 4.4. La Sección 5 se dedica al diseño de la prueba y los resultados de un estudio numérico para evaluar los diferentes métodos propuestos para resolver el RCPSP-ROC. 6 concluye este documento dando un breve resumen de los resultados y sugerencias para futuras investigaciones.

**CONCLUSIÓN**

En este artículo, presentamos una extensión del RCPSP con el costo de las horas extras y una función de ingresos disminuyendo monotónicamente con la duración del proyecto. Formalizamos el problema de programación como un programa lineal de enteros mixtos y diseñamos codificaciones como paso de preparación para el desarrollo de procedimientos de solución eficientes. Luego desarrollamos un algoritmo genético para el problema, calculado e interpretado resultados para una biblioteca de problemas basado en un conjunto de pruebas RCPSP ampliamente utilizado. Investigamos más a fondo el uso de una heurística

de búsqueda local general, ofreciendo así resultados numéricos tanto para los métodos de solución heurística específicos del problema como para los genéricos de la caja negra. Los resultados son muy prometedores. Para proyectos más grandes con muchas actividades, los métodos de solución específicos de problemas heurísticos superan a los solucionadores genéricos de cajas negras. Para proyectos de tamaño pequeño, el uso de un método heurístico de caja negra funcionó mejor.

Para futuras investigaciones, se promete utilizar operadores modificados del algoritmo genético para lograr mejores resultados, por ejemplo, el operador de cruce de pico propuesto por Valls et al. (2008). Este operador considera la idoneidad de los individuos en el crossover.

Además, la lista de actividades como núcleo de todas las representaciones evaluadas puede intercambiarse con otra codificación generalizada para inducir prioridades de actividad, la llamada representación de clave aleatoria estandarizada, Debels et al. (2006).

Sin embargo, se espera que el comportamiento general de la solución siga siendo el mismo incluso con tales mejoras. Por lo tanto, sería aún más interesante utilizar procedimientos o representaciones de solución completamente diferentes. Una codificación adecuada e idealmente más compacta puede acelerar el proceso de solución mediante la eliminación de puntos más redundantes en el espacio de solución. Una idea es evaluar una representación basada en horarios cuasi estables conocidos por problemas de nivelación de recursos con un makespan definido heurísticamente. Makespan Una vez más, el objetivo es explorar el conjunto más pequeño posible garantizado para contener una solución óptima. Sin embargo, para este fin y también para desarrollar un algoritmo exacto, sería extremadamente útil identificar y formalizar las propiedades de las soluciones óptimas.

## TRANSLATED VERSION: FRENCH

Below is a rough translation of the insights presented above. This was done to give a general understanding of the ideas presented in the paper. Please excuse any grammatical mistakes and do not hold the original authors responsible for these mistakes.

## VERSION TRADUITE: FRANÇAIS

Voici une traduction approximative des idées présentées ci-dessus. Cela a été fait pour donner une compréhension générale des idées présentées dans le document. Veuillez excuser toutes les erreurs grammaticales et ne pas tenir les auteurs originaux responsables de ces erreurs.

## INTRODUCTION

De nombreux modèles et procédures pour les problèmes de planification des projets limités par les ressources (RCPSP) supposent que les capacités des ressources renouvelables nécessaires à l'exécution des activités du projet sont exogènes et que l'objectif est de trouver un calendrier (faisable) avec un projet minimal. En réalité, les ressources renouvelables comme le travail humain ou les machines sont souvent temporairement affectées à un projet et des décisions sur les ressources supplémentaires ou les heures supplémentaires sont prises pour atteindre une courte durée de projet. Une courte durée de projet peut être économiquement attrayante pour différentes raisons. Prenons, par exemple, des projets de développement de logiciels ou des projets de construction pour les usines. Dans de tels cas, une durée de projet plus courte peut permettre une entrée plus tôt sur le marché. Ce désir d'obtenir une courte durée de projet peut, par exemple, conduire à des clauses de pénalité contractuelle ou à d'autres régimes incitatifs qui relient les paiements réels à la durée du projet. Dans de tels cas, les revenus d'un projet diminuent généralement à mesure que sa durée augmente. Cela conduit immédiatement à la question de savoir comment utiliser les

heures supplémentaires et comment planifier de tels projets avec des contraintes de capacité flexibles et rendlesrevenus dépendants de la casserole de la manière la plus rentable.

Le reste du présent document est organisé comme suit : dans la section 2, nous décrivons les hypothèses du problème de planification de projet limité par les ressources avec les revenus spécifiques à makespanet l'option de surcapacité (RCPSP-ROC), donnons des exemples concrets, démontrons les propriétés de base du problème et de la solution, et donnons un aperçu de la littérature connexe. Dans sect. 3, nous développons un modèle de décision mathématique formel pour le RCPSP-ROC et discutons des propriétés des solutions qui guident le développement des procédures de solution. Les justifications de conception et les descriptions détaillées des différentes encodages de solution pour ce problème sont indiquées dans la section 4.2. Sur cette base, nous proposons divers algorithmes génétiques et procédures de recherche locales dans les sectes. 4.3 et 4.4. La section 5 est consacrée à la conception des essais et aux résultats d'une étude numérique visant à évaluer les différentes méthodes proposées pour résoudre le RCPSP-ROC. La sect. 6 conclut ce document en donnant un bref résumé des résultats et des suggestions pour les recherches futures.

## CONCLUSION

Dans ce document, nous avons présenté une extension du RCPSP avec le coût des heures supplémentaires et une fonction de revenus monotonement décroissante avec la durée du projet. Nous avons formalisé le problème de planification comme un programme linéaire mixte et conçu des codages comme étape de préparation pour le développement de procédures de solution efficaces. Nous avons ensuite développé un algorithme génétique pour le problème, calculé et interprété les résultats d'une bibliothèque de problèmes basé sur un ensemble de tests RCPSP largement utilisé. Nous avons étudié plus en détail l'utilisation d'une recherche locale générale heuristique, offrant ainsi des résultats numériques pour les deux problèmes spécifiques ainsi que génériques black-box méthodes de solution heuristique. Les résultats sont très prometteurs. Pour les grands projets avec de nombreuses activités, les méthodes de solutions heuristiques spécifiques aux problèmes battent les solutions génériques de boîte noire inexactes. Pour les projets de petite taille, l'utilisation d'une méthode heuristique boîte noire a fonctionné mieux.

Pour les recherches futures, il promet d'utiliser des opérateurs modifiés de l'algorithme génétique pour obtenir de meilleurs résultats, par exemple, l'opérateur de croisement de pointe proposé par Valls et al. (2008). Cet opérateur considère l'aptitude des personnes dans le crossover.
En outre, la liste d'activité comme noyau de toutes les représentations évaluées peut être échangée avec un autre codage généralisé pour induire des priorités d'activité, la représentation dite normalisée des clés aléatoires, Debels et al. (2006).

Cependant, on s'attend à ce que le comportement général de la solution reste le même, même avec de telles améliorations. Par conséquent, il serait encore plus intéressant d'utiliser des procédures ou des représentations de solutions entièrement différentes. Un codage de solution approprié et idéalement plus compact peut accélérer le processus de solution en supprimant plus de points redondants dans l'espace de la solution. Une idée est d'évaluer une représentation basée sur des horaires quasi-stables connus pour les problèmes de niveau des ressources avec un makespanheuristiquement défini. Encore une fois, l'objectif est d'explorer le plus petit ensemble possible garanti pour contenir une solution optimale. Cependant, à cette fin et aussi pour développer un algorithme exact, il serait extrêmement utile d'identifier et de formaliser les propriétés de solutions optimales.

## TRANSLATED VERSION: GERMAN

Below is a rough translation of the insights presented above. This was done to give a general understanding of the ideas presented in the paper. Please excuse any grammatical mistakes and do not hold the original authors responsible for these mistakes.

**ÜBERSETZTE VERSION: DEUTSCH**

Hier ist eine ungefähre Übersetzung der oben vorgestellten Ideen. Dies wurde getan, um ein allgemeines Verständnis der in dem Dokument vorgestellten Ideen zu vermitteln. Bitte entschuldigen Sie alle grammatikalischen Fehler und machen Sie die ursprünglichen Autoren nicht für diese Fehler verantwortlich.

**EINLEITUNG**

Viele Modelle und Verfahren für ressourcenbeschränkte Projektplanungsprobleme (RCPS) gehen davon aus, dass die Kapazitäten der erneuerbaren Ressourcen, die für die Durchführung der Projektaktivitäten erforderlich sind, exogen angegeben werden und dass das Ziel darin besteht, einen (machbaren) Zeitplan mit einer minimalen Projekt-Kapazität oder -Dauer zu finden. In Wirklichkeit werden die nachwachsenden Ressourcen wie menschliche Arbeit oder Maschinen oft vorübergehend einem Projekt zugeordnet und Entscheidungen über zusätzliche Ressourcen oder Überstunden getroffen, um eine kurze Projektdauer zu erreichen. Eine kurze Projektdauer kann aus unterschiedlichen Gründen wirtschaftlich attraktiv sein. Betrachten Sie beispielsweise Softwareentwicklungsprojekte oder Bauprojekte für Fabriken. In solchen Fällen kann eine kürzere Projektdauer einen früheren Markteintritt ermöglichen. Dieser Wunsch nach einer kurzen Projektlaufzeit kann z. B. Zu Vertragsstrafenklauseln oder anderen Anreizsystemen führen, die tatsächliche Zahlungen mit Projektdauern in Beziehung setzen. In solchen Fällen sinkt der Umsatz eines Projekts in der Regel mit zunehmender Laufzeit. Dies führt sofort zu der Frage, wie Überstunden zu nutzen und wie solche Projekte mit flexiblen Kapazitätsengpässen und machenspannenabhängigeneinnahmen in der profitabelsten Weise zu planen.

Der Rest dieses Beitrags ist wie folgt organisiert: In Abschnitt 2 beschreiben wir die Annahmen des ressourcenbeschränkten Projektplanungsproblems mit makespan-spezifischeneinnahmen und der Option der Überkapazität (RCPSP-ROC), geben reale Beispiele, zeigen grundlegende Problem- und Lösungseigenschaften auf und geben einen Überblick über die zugehörige Literatur. In Abschnitt 3 entwickeln wir ein formales mathematisches Entscheidungsmodell für den RCPSP-ROC und diskutieren die Eigenschaften von Lösungen, die die Entwicklung von Lösungsverfahren leiten. Die Entwurfsgründe und detaillierten Beschreibungen verschiedener Lösungscodierungen für dieses Problem sind in Abschnitt 4.2 angegeben. Auf dieser Grundlage schlagen wir verschiedene genetische Algorithmen und lokale Suchverfahren in Sekten vor. 4.3 und 4.4. Abschnitt 5 ist dem Testentwurf und den Ergebnissen einer numerischen Studie zur Bewertung der verschiedenen vorgeschlagenen Methoden zur Lösung des RCPSP-ROC gewidmet. Abschnitt 6 schließt diesen Beitrag mit einer kurzen Zusammenfassung der Ergebnisse und Vorschläge für zukünftige Forschung.

**SCHLUSSFOLGERUNG**

In diesem Beitrag haben wir eine Erweiterung des RCPSP mit Überstundenkosten und einer monoton abnehmenden Umsatzfunktion mit Projektdauer vorgestellt. Wir formalisierten das Planungsproblem als lineares Mixed-Integer-Programm und entwickelten Codierungen als Vorbereitungsschritt für die Entwicklung effizienter Lösungsverfahren. Wir entwickelten dann einen genetischen Algorithmus für das Problem, berechneten und interpretierten Ergebnisse für eine Problembibliothek basierend auf einem weit verbreiteten RCPSP-Testsatz. Wir untersuchten weiter die Verwendung einer allgemeinen lokalen Suchheuristik und boten so numerische Ergebnisse sowohl für problemspezifische als auch generische

Black-Box-heuristische Lösungsmethoden. Die Ergebnisse sind sehr vielversprechend. Bei größeren Projekten mit vielen Aktivitäten schlagen heuristische problemspezifische Lösungsmethoden generische Blackbox-Inexakt-Solver. Bei kleinen Projekten funktionierte die Verwendung einer heuristischen Black-Box-Methode am besten.

Für zukünftige Forschungen verspricht es, modifizierte Operatoren des genetischen Algorithmus zu verwenden, um bessere Ergebnisse zu erzielen, zum Beispiel den von Valls et al. (2008) vorgeschlagenen Peak-Crossover-Operator. Dieser Operator berücksichtigt die Eignung der Personen in der Crossover.

Darüber hinaus kann die Aktivitätsliste als Kern aller bewerteten Darstellungen mit einer anderen weit verbreiteten Codierung für die Induktion von Aktivitätsprioritäten, der so genannten standardisierten zufälligen Schlüsseldarstellung, Debels et al. (2006), ausgetauscht werden.

Es wird jedoch erwartet, dass das allgemeine Lösungsverhalten auch bei solchen Verbesserungen gleich bleibt. Daher wäre es noch interessanter, völlig unterschiedliche Lösungsverfahren oder Darstellungen zu verwenden. Eine geeignete und idealerweise kompaktere Lösungscodierung kann den Lösungsprozess beschleunigen, indem mehr redundante Punkte im Lösungsraum entfernt werden. Eine Idee ist die Bewertung einer Darstellung auf der Grundlage von quasi-stabilen Zeitplänen, die für Probleme mit Ressourcenniveau mit einer heuristisch definierten Makespanbekanntsind. Auch hier ist es das Ziel, das kleinstmögliche Set zu erkunden, das garantiert eine optimale Lösung enthält. Zu diesem Zweck und auch zur Entwicklung eines genauen Algorithmus wäre es jedoch äußerst hilfreich, die Eigenschaften optimaler Lösungen zu identifizieren und zu formalisieren.

**TRANSLATED VERSION: PORTUGUESE**

Below is a rough translation of the insights presented above. This was done to give a general understanding of the ideas presented in the paper. Please excuse any grammatical mistakes and do not hold the original authors responsible for these mistakes.

**VERSÃO TRADUZIDA: PORTUGUÊS**

Aqui está uma tradução aproximada das ideias acima apresentadas. Isto foi feito para dar uma compreensão geral das ideias apresentadas no documento. Por favor, desculpe todos os erros gramaticais e não responsacule os autores originais responsáveis por estes erros.

**INTRODUÇÃO**

Muitos modelos e procedimentos para problemas de agendamento de projetos limitados por recursos (rcpsps) assumem que as capacidades dos recursos renováveis necessários para a realização das atividades do projeto são exógenamente dadas e que o objetivo é encontrar um calendário (exequível) com um projeto mínimo de duração. Makespan Na realidade, os recursos renováveis, como o trabalho humano ou a maquinaria, são muitas vezes temporariamente afetados a um projeto e são tomadas decisões sobre recursos adicionais ou horas extraordinárias para alcançar uma curta duração do projeto. Uma curta duração do projeto pode ser economicamente atraente por diferentes razões. Considere, por exemplo, projetos de desenvolvimento de software ou projetos de construção para fábricas. Nesses casos, uma duração mais curta do projeto pode permitir uma entrada mais precoce no mercado. Esta vontade de alcançar uma curta duração do projeto pode, por exemplo, conduzir a cláusulas de sanções contratuais ou a outros regimes de incentivos que digam respeito aos pagamentos efetivos às durações dos projetos. Nestes casos, a receita de um projeto diminui tipicamente à medida que a sua duração aumenta. Isto leva imediatamente à questão de saber como utilizar as horas extraordinárias e como agendar tais projetos com restrições de capacidade flexíveis e fazer com que as receitas dependentes dopano da forma mais rentável. Makespan

O restante deste trabalho é organizado da seguinte forma: na Seita. 2, descrevemos os pressupostos do problema de agendamento de projetos limitados com recursos com receitas específicas de fabricaçãoe opção de sobrecapacidade (RCPSP-ROC), dar exemplos no mundo real, demonstrar propriedades básicas de problemas e soluções, e fornecer uma visão geral da literatura relacionada. Na Seita 3, desenvolvemos um modelo formal de decisão matemática para o RCPSP-ROC e discutimos propriedades de soluções que orientam o desenvolvimento de procedimentos de solução. Os raciocínios de conceção e descrições detalhadas das diferentes codificações de soluções para este problema são indicados na seita. Nesta base, propomos vários algoritmos genéticos e procedimentos de pesquisa locais em Seitas. 4.3 e 4.4. A secção 5 é dedicada ao desenho do ensaio e aos resultados de um estudo numérico para avaliar os diferentes métodos propostos para resolver o RCPSP-ROC. A seita 6 conclui este trabalho com um breve resumo dos resultados e sugestões para futuras investigações.

## CONCLUSÃO

Neste trabalho, apresentámos uma extensão do RCPSP com custos de horas extraordinárias e uma função de receitas que diminui monótonamente com a duração do projeto. Formalizamos o problema do agendamento como um programa linear misto e concebemos codificações como passo de preparação para o desenvolvimento de procedimentos de solução eficientes. Em seguida, desenvolvemos um algoritmo genético para o problema, calculamos e interpretamos resultados para uma biblioteca problemática baseada num conjunto de testes RCPSP amplamente utilizado. Investigámos ainda a utilização de uma pesquisa heurística geral local, oferecendo assim resultados numéricos tanto para métodos específicos de problemas como para métodos genéricos de solução heurística da caixa preta. Os resultados são muito promissores. Para projetos maiores com muitas atividades, os métodos de solução heurístico específicos para problemas batem os solucionadores genéricos inexatos da caixa preta. Para projetos de pequena dimensão, a utilização de um método heurístico de caixa preta funcionou melhor.

Para futuras investigações, promete utilizar operadores modificados do algoritmo genético para obter melhores resultados, por exemplo, o operador de crossover de pico proposto pela Valls et al. (2008). Este operador considera a aptidão dos indivíduos no crossover.

Além disso, a lista de atividades como núcleo de todas as representações avaliadas pode ser trocada com outra codificação generalizada para induzir prioridades de atividade, a chamada representação-chave aleatória padronizada, Debels et al. (2006).

No entanto, espera-se que o comportamento da solução geral permaneça o mesmo mesmo com tais melhorias. Por conseguinte, seria ainda mais interessante utilizar procedimentos ou representações de soluções inteiramente diferentes. Uma codificação de solução adequada e idealmente mais compacta pode acelerar o processo de solução removendo pontos mais redundantes no espaço da solução. Uma ideia é avaliar uma representação baseada em horários quase estáveis conhecidos por problemas de nivelamento de recursos com uma êmpesa heuristicamente definida. Makespan Mais uma vez, o objetivo é explorar o conjunto mais pequeno possível garantido para conter uma solução ideal. No entanto, para este fim e também para desenvolver um algoritmo exato, seria extremamente útil identificar e formalizar propriedades de soluções ideais.