Time-Optimal Scheduling of Tasks with Shared and Dynamically Constrained Energy Systems

Eero Immonen[a]
Computational Engineering and Analysis, Turku University of Applied Sciences, Joukahaisenkatu 3-5, Turku, Finland
fi

Keywords: Task Scheduling, Mixed-Integer Nonlinear Programming, Dynamical Constraints, Energy, Genetic Algorithm.

Abstract: This article addresses the minimum-time scheduling of sequential tasks requiring energy (or a similar resource) from shared, dynamically constrained systems. Practical applications of this problem include human operations with fatigue and rest cycles, among others. The goal is to jointly optimize task execution order and power allocation to the tasks, balancing execution speed with necessary recovery periods and task transition times. We present a generic Mixed-Integer Nonlinear Programming (MINLP) formulation of the problem, propose a heuristic solution method based on a Genetic Algorithm (GA), and demonstrate its use in a numerical example on efficient execution of a two-exercise workout. The numerical example shows that the proposed heuristic method rapidly produces a solution within 0.9% of the one obtained via the MINLP solver SCIP.

1 INTRODUCTION

This article addresses the minimum-time scheduling of sequential tasks that require energy — or other similar resource — from shared, dynamically constrained systems. There are many interesting practical applications of such scheduling problems. For example, a team of firefighters, through a collaborative effort completes physically demanding tasks (e.g. lifting, digging, carrying) under time constraints. Each task consumes energy, and the firefighters may need to rest and recover to be able to undertake the remaining tasks. A structurally similar problem is that of a high-performance computing system executing a set of complex computations (e.g. image processing) on multiple processors, aiming to minimize the total execution time in the presence of Ohmic heating. The computing tasks generate heat that may need to be dissipated by idling the processors before undertaking subsequent tasks. By Newton's law of cooling, such thermal recovery is not immediate.

In both above examples, besides controlling the task execution *order*, the operator(s) choose(s) the *power* used to engage in the tasks, in order to control the task execution speed: A higher power yields a faster task execution time but may later require a rest-recovery that, on the other hand, causes a delay. Such task scheduling is thus a tradeoff between local and global efficiency. Typically, in practice, there are also transition times between the tasks, and they are not necessarily symmetric, i.e. task transition $A \to B$ takes more time than $B \to A$ (e.g. climbing stairs up vs. down).

The purpose of this article is to introduce a generic mathematical (MINLP) formulation of this problem, address its heuristic numerical solution by a GA, and provide a numerical example to illustrate the framework.

2 RELATION TO LITERATURE

Several research articles have addressed time-optimal task scheduling involving *energy consumption*, see e.g. the survey papers (Ghafari et al., 2022) and (Bambagini et al., 2016) and the references therein. However, while in these articles the *power* is a dynamical design variable, the objective is to minimize total energy consumption. Consequently, the *energy system* is assumed to be a *static* resource whose size is to be minimized for a given set of tasks. As such, this research typically has targeted an efficient technological design, such as low-energy cloud computing environment, whereas in the present article we are mainly interested in efficient human operation.

Efficient human operation is also addressed in the vast literature on staff rostering (see e.g. (Ngoo et al., 2022)) and project scheduling (see e.g. (Sánchez

[a] https://orcid.org/0000-0001-5690-287X

Immonen, E.
Time-Optimal Scheduling of Tasks with Shared and Dynamically Constrained Energy Systems.
DOI: 10.5220/0013659100003982

In *Proceedings of the 22nd International Conference on Informatics in Control, Automation and Robotics (ICINCO 2025) - Volume 1*, pages 169-175
ISBN: 978-989-758-770-2; ISSN: 2184-2809

et al., 2023)). Such research addresses efficient allocation of finite, static and potentially irreplaceable resources. These resources are typically equipment or workforce, but optimal project scheduling with respect to (static) green project indicators (GPIs) i.e., energy, noise, and safety, has also been studied (Rahman et al., 2022). On the other hand, those articles that address optimal project scheduling under *dynamical* resource constraints typically treat them as binary variables, i.e. disturbances to resource availability; see e.g. (Xu and Bai, 2024).

In the present article, the objective is to minimize the execution time of a sequence of tasks constrained by dynamical energy systems. These energy systems are described by differential equations that arise from the seminal theory of human endurance from the early 1970s. We emphasize, though, that the same equations can also represent other physical systems (see Subsection 3.3). This mathematical framework (Keller, 1973), see also (Pritchard, 1993), for optimal running is an elegant mix of force balance (for locomotion) and power balance (for metabolism) considerations. Indeed, Keller's model was able to predict the prevailing world record running times for various distances with good accuracy. However, whereas that model attempts to predict the optimal race times on flat unidirectional tracks, in the present article we adapt it to scheduling of different tasks. Moreover, this article addresses multiple interconnected energy systems, whereas Keller's model only has one.

Among those few published articles that, similar to this paper, address task scheduling under dynamical constraints arising from differential equations, we mention the work of Zhou et al. (Zhou et al., 2015). They studied the minimization of energy consumption of multiprocessor system-on-chip in a schedule duration, under the constraints of real-time task deadlines and temperature limit. Their work builds on physics-based thermal modeling using lumped-parameter systems. The scheduling problem we consider in this article is constrained by dynamical energy systems with recovery, and, instead of minimizing energy consumption, we aim at a minimum-time schedule while maintaining a nonnegative energy in all systems at all times.

The problem of time-optimal scheduling of tasks with shared and dynamically constrained energy systems is formulated in this article as a MINLP. To solve it, we propose a heuristic GA-based method. GAs have been found effective for MINLPs because they can efficiently explore large, non-convex, and discontinuous search spaces without requiring gradient information (Yang, 2020). Their population-based evolution, through controlled mutation and crossover operations, allows handling discrete and continuous variables simultaneously, making them efficient for the combinatorial and nonlinear structure of MINLPs. Over the course of the past decades, GAs have been successfully used in solving task scheduling problems, including those with energy considerations (see (Pirozmand et al., 2021) and the references therein).

3 PROBLEM FORMULATION

The optimization problem addressed in this paper involves finding the fastest execution sequence (schedule) for a set of tasks. The time to complete each task depends on the chosen power, which consumes one or more dynamical pools of energy.

Although the problem has time-dependent features — early decisions influence the future state of the energy systems — it is formulated in the present section as a finite-horizon MINLP with nonlinear state-update constraints. This formulation enables encoding discrete task ordering, continuous power allocation, and dynamical resource evolution within a single mathematical system (5). The problem displays aspects of discrete-time optimal control, but due to the presence of both integer and nonlinear constraints, it is perhaps best described as an MINLP.

3.1 Definitions

Let $i \in \{1,\dots,N\}$ index the tasks t_i. Let $k \in \{1,\dots,n\}$ index the positions in the task execution sequence, with $n = \sum_{i=1}^{N} M_i$, where $M_i > 0$ is the prescribed total number of tasks of type i in the schedule. Each task t_i, $i \in \{1,\dots,N\}$, requires energy $E_i > 0$ to be completed.

The *discrete* decision variables (to be optimized) are the task execution order $x_{k,i}$, with:

$$x_{k,i} = \begin{cases} 1, & \text{if the task at position } k \text{ is of type } i, \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

such that precisely one task is executed at every position, i.e. $\sum_{i=1}^{N} x_{k,i} = 1, \forall k = 1,\dots,n$, and all required repetitions are carried out, i.e. $\sum_{k=1}^{n} x_{k,i} = M_i, \forall i = 1,\dots,N$. The *continuous* decision variables (also to be optimized) are the task execution powers $p_k \in [p_{\min}, P_{\max}]$, with $0 < p_{min} < P_{max} < \infty$, $k = 1,\dots,n$.

At each position k, the chosen power p_k yields the execution time d_k as:

$$d_k = \sum_{i=1}^{N} \frac{E_i}{p_k} x_{k,i} \tag{2}$$

Thus, if the task at schedule position k is of type i (i.e. $x_{k,i} = 1$), then the task execution time is $d_k = \frac{E_i}{p_k}$.

Transitioning from task t_{i_1} to task t_{i_2} incurs a delay, as given by the matrix:

$$T(i_1,i_2) \geq 0, \quad i_1,i_2 = 1,\ldots,N. \tag{3}$$

so that the total execution time of the schedule is $\sum_{k=1}^{n} d_k + \sum_{k=1}^{n-1} T(i_k,i_{k+1})$, which is to be minimized.

The choice of power p_k is constrained by J dynamical energy systems, indexed by $j \in \{1,\ldots,J\}$. Each energy system j is updated from the conclusion of task position $k-1$ to the conclusion of the subsequent position $k > 0$ based on the chosen power p_k:

$$L_{j,k} = \min\left\{ L_{j,k-1} + d_k\left(\sigma_j - \sum_{i\in I} c_{i,j}\, x_{k,i}\, p_k\right),\, L_{\max,j} \right\}, \tag{4}$$

where σ_j is the scalar recovery rate for the energy system j, $0 < L_{\max,j} < \infty$ is the maximum energy content of system j, and $0 \leq c_{i,j} < \infty$ is the energy drain coefficient for task i on energy system $j \in J$. Note that the completion of a task t_i can require energy from more than one system j (as in collaborative effort). Initially $L_{j,0} > 0$, $\forall j \in J$, and we require all energy systems to remain non-negative at all times:

$$L_{j,k} \geq 0, \quad \forall j \in J,\ k = 1,\ldots,n.$$

3.2 Optimization Problem

With the definitions given in Subsection 3.1, the full optimization problem is:

$$\min_{p_k, x_{k,i}} \quad \sum_{k=1}^{n} d_k + \sum_{k=1}^{n-1} T(i_k,i_{k+1}) \tag{5a}$$

$$s.t. \quad d_k = \sum_{i=1}^{N} \frac{E_i}{p_k}\, x_{k,i}, \quad k = 1,\ldots,n, \tag{5b}$$

$$L_{j,0} > 0, \quad \forall j \in J, \tag{5c}$$

$$L_{j,k} = \min\left\{ L_{j,k-1} + d_k\left(\sigma_j - \sum_{i=1}^{N} c_{i,j}\, x_{k,i}\, p_k\right), \right.$$
$$\left. L_{\max,j} \right\}, \quad \forall j \in J,\ k = 1,\ldots,n, \tag{5d}$$

$$L_{j,k} \geq 0, \quad \forall j \in J,\ k = 1,\ldots,n, \tag{5e}$$

$$\sum_{i=1}^{N} x_{k,i} = 1, \quad x_{k,i} \in \{0,1\}, \quad k = 1,\ldots,n, \tag{5f}$$

$$\sum_{k=1}^{n} x_{k,i} = M_i, \quad \forall i = 1,\ldots,N, \tag{5g}$$

$$p_k \in [p_{\min}, P_{\max}], \quad k = 1,\ldots,n. \tag{5h}$$

3.3 On the Energy System Model

The energy system model in Keller's theory of competitive running (Keller, 1973) is:

$$\frac{dE}{dt} = \sigma - P, \quad E(0) = E_0 > 0 \tag{6}$$

where $\sigma > 0$ denotes the constant recovery rate and, by definition, the running power $P = fv$, i.e. force times velocity. Equation (4) is a discrete-time analogy of Equation (6), obtained via a simple Euler integration (though the integration time is a variable to be optimized). In Equation (4), also recovery beyond a finite maximum value is prohibited. Moreover, contrary to Keller's model, in Equation (4), there is not necessarily a $1-1$ correspondence between the tasks and energy systems.

It is important to highlight that the energy Equation (6) has structurally similar analogs in many other physical systems. For example, by so-called Coulomb counting, the State-of-Charge (SoC) of an electric vehicle Lithium-Ion battery can be represented by (Immonen and Hurri, 2021):

$$\frac{d\mathrm{SoC}}{dt} = r - \frac{I}{Q_n}, \quad \mathrm{SoC}(0) \in (0,1] \tag{7}$$

where r is a regeneration rate, $I \geq 0$ is the discharge current and Q_n is the nominal battery capacity. Clearly, Equation (7) is Equation (6) for $E = \mathrm{SoC}$, $\sigma = r$ and $P = I/Q_n$, with I as the design variable.

On the other hand, the Ohmic (Joule) heating of an electric circuit can be modeled by:

$$\frac{dT_c}{dt} = \lambda(T_{amb} - T_c) + RI^2, \quad T_c(0) = T_0 \tag{8}$$

where $R > 0$ is resistance, I is electric current, $T_{amb} - T_c$ is the temperature difference between ambient (amb) and the circuit (c), and $\lambda > 0$ is a coefficient of heat transfer. Then Equation (8) is just Equation (6) for $E = T_c$, $\sigma = -\lambda T_c$ and $P = -\lambda T_{amb} - RI^2$, with I as the design variable. The optimization constraint would be maintaining $T_c(t) \leq T_{max}\ \forall t \geq 0$.

In Equation (7), r is typically dependent on time (or driving profile or terrain). In Equation (8), the power term P also has a disturbance (λT_{amb}) and the recovery term σ is not a constant but involves state feedback from E. Such features may complicate the numerical solution of Problem (5) under these dynamical constraints. The author expects to address them in a future article.

4 HEURISTIC SOLUTION ALGORITHM

In the Appendix of this article, we describe a GA that, based on numerical experiments, rapidly yields reasonably good solutions to Problem (5). The method is described in Algorithm 1, which is based on the two supplementary methods, Algorithm 2 (mutation) and Algorithm 3 (crossover). The algorithm implementations follow the typical structure for genetic algorithms. A slight added complexity arises from ensuring that the population satisfies the constraints (5f)-(5g) at all times.

5 NUMERICAL EXAMPLE

5.1 Problem Description

Let us consider, as a simple numerical example, the optimal execution of a physical workout consisting of two exercises. We seek to determine the minimum-time execution sequence, and the corresponding powers, for performing $M_1 = 10$ squats and $M_2 = 10$ push-ups, i.e. $i \in \{1, 2 = N\}$ and $k \in \{1, \ldots, 20 = n\}$.

Each exercise type ($i = 1$ for squat, $i = 2$ for push-up) requires a fixed amount of energy E_i per repetition. Based on the change of potential energy, for a 80 kg person, one squat is assumed to consume $E_1 = 408$ J, corresponding to approximately 0.52 m vertical movement. Similarly, experimental research reports that standard push-ups use between 69% and 75% of body mass (Ebben and Jensen, 1998). Assuming 71% of 80 kg body mass lifted over 0.45 m, we obtain the energy consumption $E_2 = 250$ J for a single push-up.

Each exercise transition incurs a time penalty, defined by the transition matrix T:

$$T = \begin{bmatrix} 0.5 & 2.5 \\ 5 & 0.5 \end{bmatrix} \tag{9}$$

where going up from push-up position to squat position $T(2,1)$ takes longer (5 s) than the reverse transition $T(1,2)$ due to gravity. Repeating either exercise takes 0.5 s, and it is thus always faster than switching to the other exercise.

Based on the major muscles activated in the two workout exercises considered, we assume that the human body has two energy systems, one for the lower body ($j = 1$) and the other for the upper body ($j = 2$). Both exercises consume energy from both systems but at different rates, per the drain coefficient matrix c:

$$c = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1 \end{bmatrix} \tag{10}$$

This indicates that a squat (resp. push-up) primarily consumes energy from the lower (resp. upper) body energy system, but it also means doing an exercise makes it more challenging to immediately thereafter do any exercise. This is consistent with practical observations from human endurance training.

We assume that the recovery rates are $\sigma_1 = \sigma_2 = 60$ J/s, and that initially the person is fully recovered, with $L_0 = L_{\max} = [1100, 400]$ J. Finally, we assume that the maximum power capacity for this individual is $P_{max} = 200$ W, representing a single muscle group estimate for an untrained adult (McBride et al., 1999).

5.2 SCIP Global Optimum Solution

To obtain a reference solution to the two-exercise scheduling problem, the SCIP Optimization Suite 9.0 MINLP solver (Bolusani et al., 2024) was executed for 2 hours on the CSC Puhti computing environment. The best feasible schedule found has a total duration of 78.5 s, with the primal-dual optimality gap at 0% indicating the global optimum. During execution, SCIP explored approximately 6.59 million nodes, generated 638 feasible solutions, and consumed 10 GB of memory.

5.3 Genetic Algorithm Solution

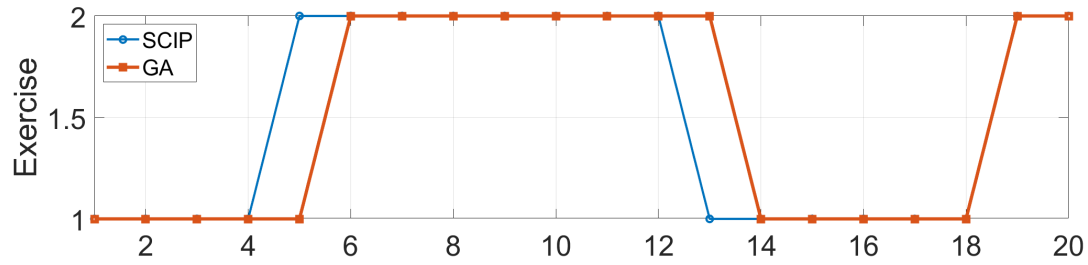
The proposed GA was implemented in Python 3.11 and executed on a laptop workstation with 12th Generation Intel(R) Core(TM) i7-1265U processor and 32 GB memory. With $S_p = 100$ (population size), $G = 3000$ (number of generations), $\mu = 0.2$ (mutation rate) and $\rho = 0.05$ (power change rate), the code execution completed in less than 2 minutes. The best execution sequence found by the GA is:

$$\{\underbrace{1,1,1,1,1}_{\text{Squats}},\underbrace{2,2,2,2,2,2,2,2}_{\text{Push-ups}},\underbrace{1,1,1,1,1}_{\text{Squats}},\underbrace{2,2}_{\text{Push-ups}}\}$$

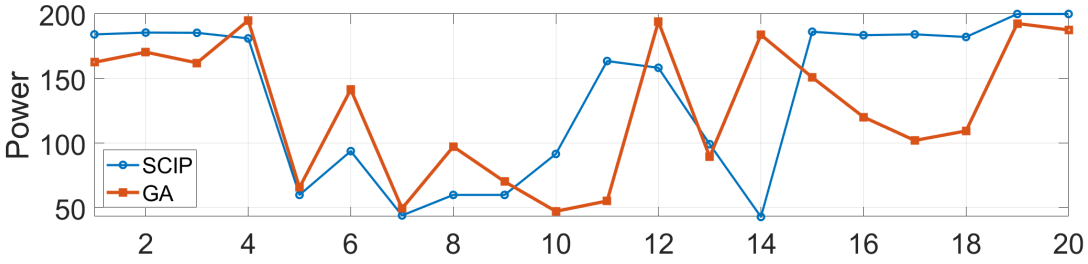
with a total execution time of 79.2 s.

5.4 Comparison and Discussion

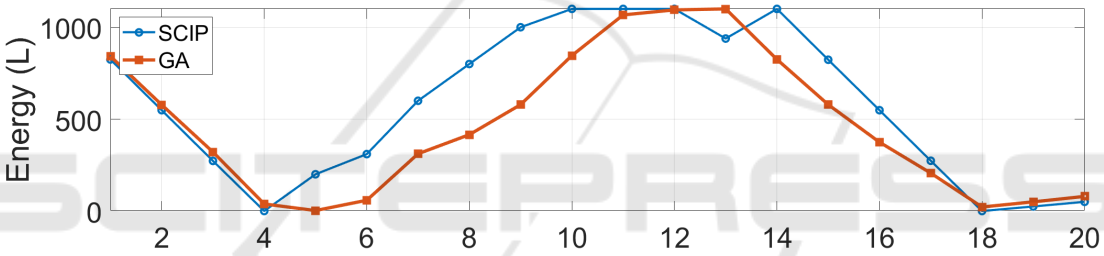
Figure 1 presents a detailed comparison of the optimal solutions obtained by SCIP (Subsection 5.2) and the proposed GA (Subsection 5.3). Though not identical, the solutions are very similar: Both favor repetitions of the same exercise once started, to exploit faster transitions. Moreover, as demonstrated in Figure 1c and Figure 1d, both schedules first drain all the energy from the lower body system, then recover that system to full during 8 push-ups that fully drain the upper body energy system. The lower body energy system is then again fully drained in squats, and both



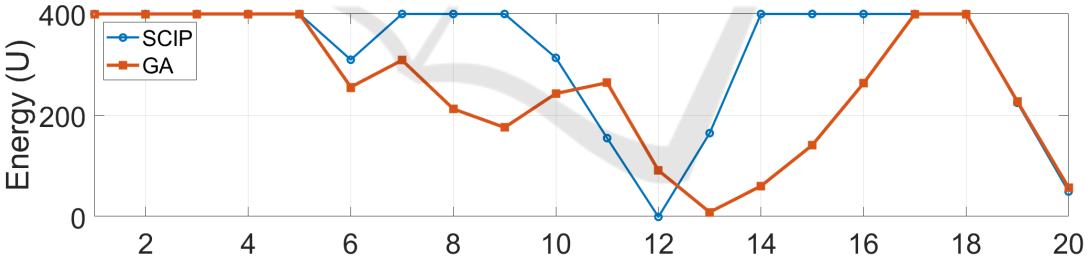
(a) Optimal schedule (1 = Squat, 2 = Push-up)



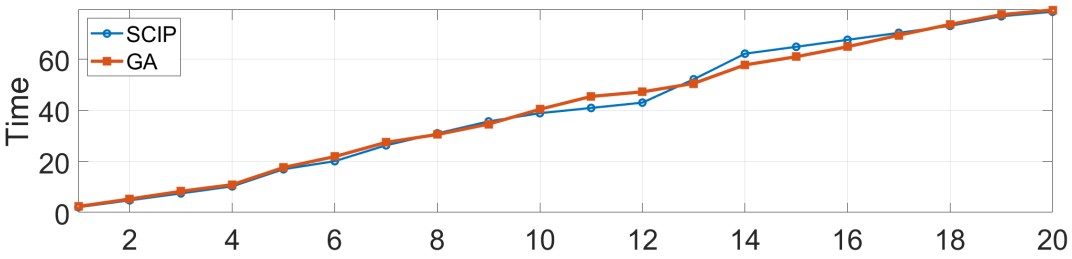
(b) Power allocation in exercises



(c) Energy in lower body after each exercise



(d) Energy in upper body after each exercise



(e) Cumulative execution time (s)

Figure 1: Comparison of the optimal solutions from SCIP and GA for the numerical example in Section 5. In all plots, the horizontal axis denotes the task position (k).

optimal plans finish with the remaining 2 push-ups at near-maximal power. The cumulative execution time profiles (Figure 1e) are almost identical, except between positions 11-16.

The schedule obtained from the proposed GA is suboptimal. However, the GA solution is obtained rapidly in comparison to SCIP and the fastest GA schedule is within 0.9% of the SCIP global minimum. Moreover, the total energy expenditures corresponding to the power assignment schedules of Figure 1b are: 6579.78 J (SCIP) and 6580.0 J (GA). This very small difference of 0.003% shows that the SCIP global optimum power plan is not substantially more energy efficient either. The proposed GA thus appears to provide a promising alternative for rapid generation of high-quality solution candidates for Problem (5).

The workout example addressed herein is perhaps contrived — it merely aims to illustrate the proposed GA for solving Problem (5). However, there are well-known fitness workouts such as *Angie* in crossfit (BarBend, 2023) with a similar structure. Solving the corresponding optimization problems, as in this section, would thus also have practical significance in sports training. Indeed, such optimization results could be used to design the best strategy for a competition. They could also be utilized in training to determine whether improvement in total execution time is due to increased fitness or just more clever planning of the workout execution.

6 CONCLUSIONS AND FUTURE WORK

This article has addressed the minimum-time scheduling of sequential tasks that consume energy from shared, dynamically constrained systems. We have presented a general MINLP formulation of the problem, developed a heuristic solution method based on a GA, and demonstrated its application, with good performance related to an off-the-shelf solver SCIP, in a numerical example involving a two-exercise workout.

Perhaps the most interesting future applications of the mathematical framework presented in this article are collaborative human scheduling problems, such as emergency teams. In the future, it also is important to address minimum-time scheduling problems with more complex dynamical constraints. As discussed in Subsection 3.3, one such problem is thermal management where the dynamical system involves state feedback. The formalism presented in this article is easy to adapt to the new domain, but efficient solution may require further adoption of optimal control methods.

REFERENCES

Bambagini, M., Marinoni, M., Aydin, H., and Buttazzo, G. (2016). Energy-aware scheduling for real-time systems: A survey. *ACM Transactions on Embedded Computing Systems (TECS)*, 15(1):1–34.

BarBend (2023). How to do the Angie workout in crossfit? Available at: https://barbend.com/crossfit-angie-workout/ (Accessed: 2025-04-03).

Bolusani, S., Besançon, M., Bestuzheva, K., Chmiela, A., Dionísio, J., Donkiewicz, T., van Doornmalen, J., Eifler, L., Ghannam, M., Gleixner, A., et al. (2024). The scip optimization suite 9.0. *arXiv preprint arXiv:2402.17702*.

Ebben, W. P. and Jensen, R. L. (1998). Strength training for women: Debunking myths that block opportunity. *The Physician and sportsmedicine*, 26(5):86–97.

Ghafari, R., Kabutarkhani, F. H., and Mansouri, N. (2022). Task scheduling algorithms for energy optimization in cloud environment: a comprehensive review. *Cluster Computing*, 25(2):1035–1093.

Immonen, E. and Hurri, J. (2021). Incremental thermoelectric cfd modeling of a high-energy lithium-titanate oxide battery cell in different temperatures: A comparative study. *Applied Thermal Engineering*, 197:117260.

Keller, J. B. (1973). A theory of competitive running. *Physics today*, 26(9):42–47.

McBride, J. M., Triplett-McBride, T., Davie, A., and Newton, R. U. (1999). A comparison of strength and power characteristics between power lifters, olympic lifters, and sprinters. *The Journal of Strength & Conditioning Research*, 13(1):58–66.

Ngoo, C. M., Goh, S. L., Sabar, N. R., Abdullah, S., Kendall, G., et al. (2022). A survey of the nurse rostering solution methodologies: The state-of-the-art and emerging trends. *IEEE Access*, 10:56504–56524.

Pirozmand, P., Hosseinabadi, A. A. R., Farrokhzad, M., Sadeghilalimi, M., Mirkamali, S., and Slowik, A. (2021). Multi-objective hybrid genetic algorithm for task scheduling problem in cloud computing. *Neural computing and applications*, 33:13075–13088.

Pritchard, W. G. (1993). Mathematical models of running. *Siam review*, 35(3):359–379.

Rahman, H. F., Chakrabortty, R. K., Elsawah, S., and Ryan, M. J. (2022). Energy-efficient project scheduling with supplier selection in manufacturing projects. *Expert Systems with Applications*, 193:116446.

Sánchez, M. G., Lalla-Ruiz, E., Gil, A. F., Castro, C., and Voß, S. (2023). Resource-constrained multi-project scheduling problem: A survey. *European Journal of Operational Research*, 309(3):958–976.

Xu, J. and Bai, S. (2024). A reactive scheduling approach for the resource-constrained project scheduling problem with dynamic resource disruption. *Kybernetes*, 53(6):2007–2028.

Yang, X.-S. (2020). *Nature-inspired optimization algorithms*. Academic Press.

Zhou, J., Wei, T., Chen, M., Yan, J., Hu, X. S., and Ma, Y. (2015). Thermal-aware task scheduling for energy

minimization in heterogeneous real-time mpsoc systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(8):1269–1282.

APPENDIX

Algorithm 1: Genetic algorithm for heuristic solution of Problem (5).

Parameters: S_p (population size), G (number of generations), $\mu > 0$ (mutation rate), $\rho > 0$ (power change rate).;
Data: $i \in \{1,\dots,N\}$, M_i, E_i, σ_j, $c_{i,j}$, $T(i_1,i_2)$, $[p_{\min}, P_{\max}]$.
Result: $x^* = \{x^*_{k,i}\}$ and $p^* = \{p^*_k\}$ that minimize $f(x,p) = \sum_{k=1}^{n} d_k + \sum_{k=1}^{n-1} T(i_k, i_{k+1})$ subject to Equations (5b)-(5h).
Initialization: Generate initial population $(x,p) \in \mathcal{P}$ such that $\text{SIZE}(\mathcal{P}) = G$ and:
- $x = \{x_{k,i}\}$ satisfies $\sum_{i=1}^{N} x_{k,i} = 1, \forall k = 1,\dots,n$, and $\sum_{k=1}^{n} x_{k,i} = M_i, \forall i = 1,\dots,N$.
- $p = \{p_k\}$ where p_k is uniformly randomized from $[p_{\min}, P_{\max}]$.

Initialize $f^* = \infty$.;
for $g = 1$ **to** G **do**
  Initialize a new population $\mathcal{P}_{\text{new}} = \emptyset$.;
  while $\text{SIZE}(\mathcal{P}_{new}) < S_p$ **do**
    Parent selection: Randomly select two pairs (4 candidates) from $\mathcal{P}$ and choose, from each pair, the candidate with lower $f(x,p)$ as parents, denoted by (x^1,p^1) and (x^2,p^2).;
    Crossover:
    $(x^c,p^c) \leftarrow \text{CROSSOVER}\Big((x^1,p^1),(x^2,p^2)\Big)$
    mutation:
    $(x',p') \leftarrow \text{MUTATION}\big((x^c,p^c),\mu,\rho,p_{min},P_{\max}\big)$
    Add (x',p') to $\mathcal{P}_{\text{new}}$.;
  end
  $\mathcal{P} \leftarrow \mathcal{P}_{\text{new}}$.;
  Compute $f(x,p)$ for all $(x,p) \in \mathcal{P}$.;
  if $\min_{(x,p)\in\mathcal{P}} f(x,p) < f^*$ **then**
    $(x^*,p^*) \leftarrow \arg\min_{(x,p)\in\mathcal{P}} f(x,p)$.;
    $f^* = \min_{(x,p)\in\mathcal{P}} f(x,p)$.
  end
end
return (x^*,p^*) and f^*.

Algorithm 2: Mutation operation for Algorithm 1.

Data: $(x,p) \in \mathcal{P}$, $\mu > 0$, $\rho > 0$, $[p_{min}, P_{max}]$.
Result: Mutated candidate solution (x',p').
Set $x' \leftarrow x$ and $p' \leftarrow p$.;
$r \leftarrow \text{rand}([0,1])$.;
if $r < \mu$ **then**
  $k_1, k_2 \leftarrow \text{rand}(\{1,\dots,n\}), k_1 \neq k_2$.;
  Swap $x'_{k_1,i} \longleftrightarrow x'_{k_2,i}, \forall i \in 1,\dots,N$.;
end
for $k = 1$ **to** n **do**
  $r \leftarrow \text{rand}([0,1])$.;
  if $r < \mu$ **then**
    $\delta \leftarrow \text{rand}([-\rho P_{max}, \rho P_{max}])$.;
    $p'_k \leftarrow \max\big(p_{\min}, \min(p'_k + \delta, P_{\max})\big)$.;
  end
end
return (x',p').

Algorithm 3: Crossover operation for Algorithm 1.

Data: Two parent solutions (x^1,p^1) and (x^2,p^2).
Result: Child solution (x^c,p^c).
$r \leftarrow \text{rand}(\{1,\dots,n-1\})$.;
For each $k = 1,\dots,n$, set

$$x^c_{k,i} \leftarrow \begin{cases} x^1_{k,i}, & k \le r, \\ x^2_{k,i}, & k > r, \end{cases} \quad \forall i = 1,\dots,N.$$

Let $C_i \leftarrow \sum_{k=1}^{n} x^c_{k,i}$ $\forall i = 1,\dots,N$.;
while $\exists i \in \{1,\dots,N\}$ *with* $C_i > M_i$ **do**
  foreach $i \in \{1,\dots,N\}$ *such that* $C_i > M_i$ **do**
    $D_i \leftarrow C_i - M_i$.;
    for $d = 1$ **to** D_i **do**
      $k \leftarrow \text{rand}(\{\, k \in \{1,\dots,n\} \mid x^c_{k,i} = 1 \,\}$.;
      $j \leftarrow \text{rand}(\{\, j \in \{1,\dots,N\} \mid C_j < M_j \,\}$.;
      $x^c_{k,i} \leftarrow 0, x^c_{k,j} \leftarrow 1$.;
      $C_i \leftarrow C_i - 1, C_j \leftarrow C_j + 1$.;
    end
  end
end
$r_2 \leftarrow \text{rand}(\{1,\dots,n-1\})$.;
For each $k = 1,\dots,n$, set

$$p^c_k \leftarrow \begin{cases} p^1_k, & k \le r_2, \\ p^2_k, & k > r_2. \end{cases}$$

return (x^c,p^c).