XIV International Conference "Optimization and Applications"
(OPTIMA-2023)

# On Solving the Robust Transfer Line Balancing Problem with Parallel Tasks and Interval Processing Times

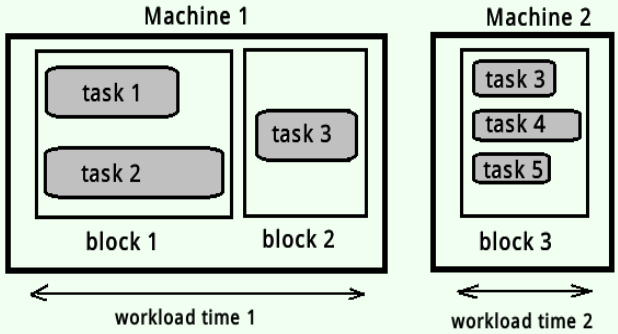
**Pavel Borisovsky**
*Sobolev Institute of Mathematics SB RAS (Omsk department), Omsk, Russia*

**This research is supported by Russian Science Foundation grant 22–71–10015**

# Problem formulation

## Transfer line structure

- A set of *tasks* is given. Each task must be performed once.
- Tasks are grouped into *blocks* for a parallel execution. The block time is the maximal execution time of its tasks.
- Each *machine* can execute a sequence of blocks. A *workload time* of a machine is the sum of its blocks' times and it is limited by a given *cycle time* $\boldsymbol{T}$.
- A partial order on the tasks is given that defines *precedence* relations.



# Literature Review

Deterministic problem

**1.** Dolgui, A., Guschinsky, N., Levin, G.: On problem of optimal design of transfer lines with parallel and sequential operations. In: 7th IEEE International Conference on ETFA-1999, vol. 1, pp. 329–334.

Problem formulation, motivation, graph solution approach.

**2.** Guschinskaya, O., Gurevsky E., Dolgui, A., Eremeev, A.: Metaheuristic approaches for the design of machining lines. Int. J. Adv. Manuf. Technol **55**(1-4), 11–22 (2011)

GRASP, Genetic algorithm, test instances.

# Robust problem formulation

$\boldsymbol{V = \{1, 2, ..., n\}}$ is the set of all tasks;

$\boldsymbol{t = (t_1, ..., t_n)}$ is the vector of nominal execution times;

$\boldsymbol{\widetilde{V} \subseteq V}$ is the set of *uncertain* tasks, for which the execution times may deviate from their nominal values;

# Robust problem formulation

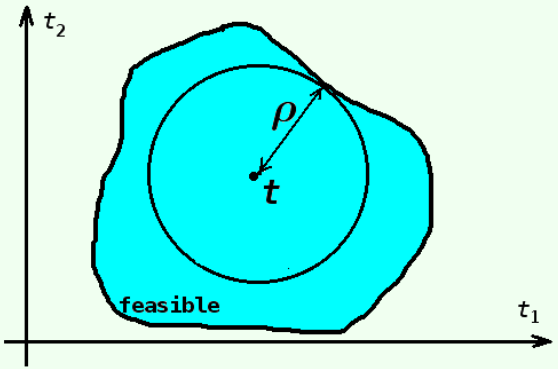
For some solution $\boldsymbol{S}$, a *stability radius* is the deviation supported by the solution which keeps its feasibility.

$$\boldsymbol{\rho(S,t) = \max\{\varepsilon \mid \forall \xi \in B(\varepsilon)} \text{ solution } \boldsymbol{S} \text{ stays feasible if}$$

$$\boldsymbol{t} \text{ is replaced by } \boldsymbol{t + \xi\}.}$$

where

$$\boldsymbol{B(\varepsilon) = \{\xi \in R^n \mid \xi_j > 0} \text{ for uncertain tasks } \boldsymbol{j,} \text{ and } \boldsymbol{\|\xi\| \leq \varepsilon\}}$$




In this work, $\boldsymbol{l_1}$-norm is used: $\boldsymbol{\|\xi\|_1 = \sum_j \xi_j.}$

# Robust problem formulation

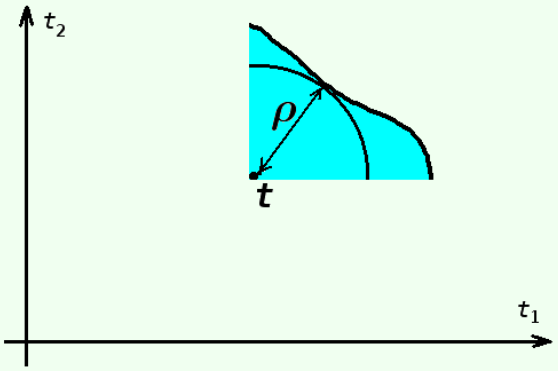
For some solution $\boldsymbol{S}$, a *stability radius* is the deviation supported by the solution which keeps its feasibility.

$$\boldsymbol{\rho(S,t) = \max\{\varepsilon \mid \forall \xi \in B(\varepsilon)}\ \text{solution}\ \boldsymbol{S}\ \text{stays feasible if}$$

$$\boldsymbol{t}\ \text{is replaced by}\ \boldsymbol{t+\xi\}.}$$

where

$$\boldsymbol{B(\varepsilon) = \{\xi \in R^n \mid \xi_j > 0}\ \text{for uncertain tasks}\ \boldsymbol{j},\ \text{and}\ \boldsymbol{\|\xi\| \le \varepsilon\}}$$




In this work, $\boldsymbol{l_1}$-norm is used: $\boldsymbol{\|\xi\|_1 = \sum_j \xi_j.}$

# Literature Review

**1.** Pirogov A., Gurevsky E., Rossi A., Dolgui A.: Robust balancing of transfer lines with blocks of uncertain parallel tasks under fixed cycle time and space restrictions. Eur. J. Oper. Res **290**(3), 946–955 (2021)

Robust problem formulation, calculation of stability radius, MIP, test instances, greedy algorithm.

**2.** Borisovsky P., Battaïa O.: MIP-Based Heuristics for a Robust Transfer Lines Balancing Problem. In: OPTIMA 2021, LNCS, vol. 13078, pp. 123–135.

Inclusion/exclusion constraints, MIP based matheuristic.

# Research objectives

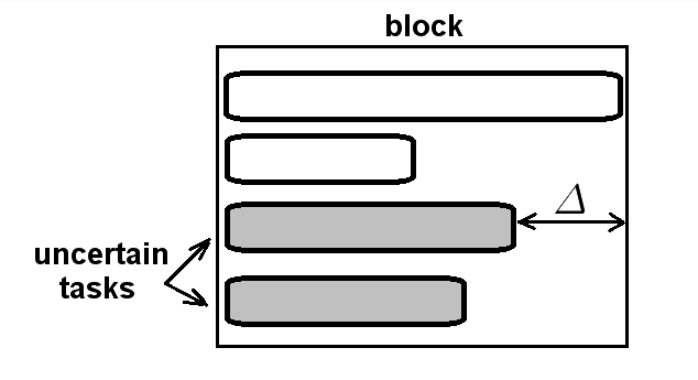
**1.** Derive a formula to compute the stability radius.

**2.** Develop an algorithm to find a feasible solution maximizing the stability radius.

### Calculation of stability radius

For any uncertain block $\boldsymbol{k}$ of machine $\boldsymbol{p}$ define a *save time* as the difference between the block working time $\boldsymbol{\tau_k}$ and the nominal processing time of its longest uncertain task, *i.e.*,

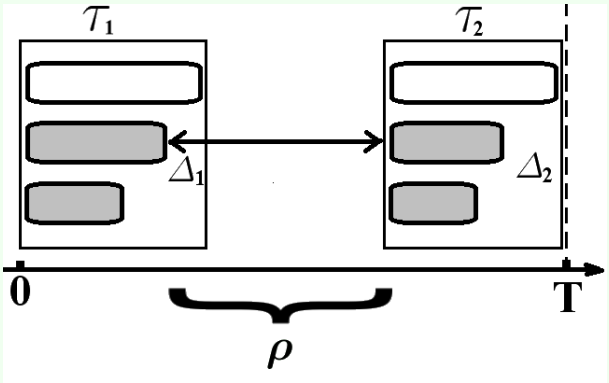
$$\boldsymbol{\Delta := \tau - \max_{j \in \widetilde{V}} t_j.}$$



## Calculation of stability radius, [1]

$$\boldsymbol{\rho = T - \sum \tau_k + \Delta_{min}}$$

i.e, take the block with the minimal $\boldsymbol{\Delta}$ and extend its longest time until the total workload reaches $\boldsymbol{T}$.




For a line with many machines $\boldsymbol{\rho = \min \rho_i}$.

---

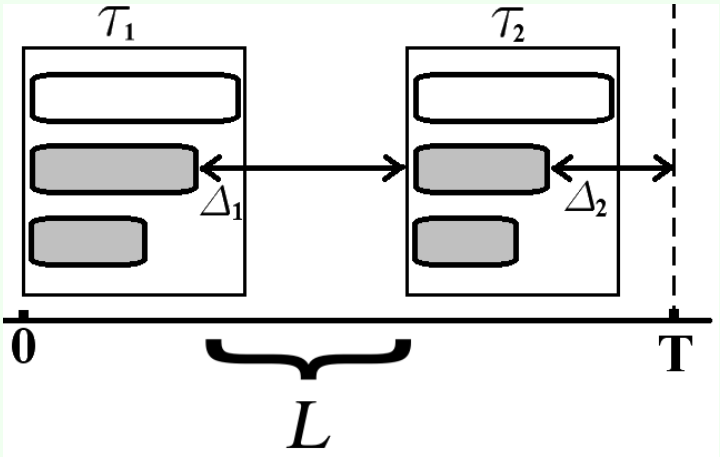
1. Pirogov A., Gurevsky E., Rossi A., Dolgui A.: Robust balancing of transfer lines with blocks of uncertain parallel tasks under fixed cycle time and space restrictions. EJOR **290**(3), 946–955 (2021)

# New formulation: interval processing times

In the previous formulation, no upper limit on the possible extra times was assumed.

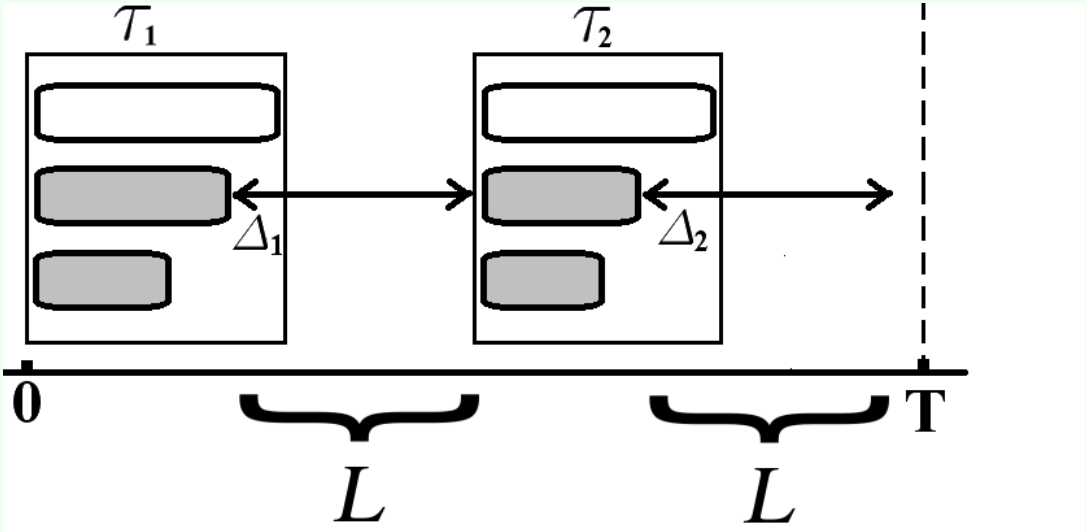
This may lead to improbable situations, where only one task has a large extra time: $\boldsymbol{\xi = (0, .., \rho, ..0)}$

**New problem formulation.**
In this study, assume that $\boldsymbol{\xi_j \leq L}$, i.e. the actual time is in $\boldsymbol{[t_j, t_j + L]}$.



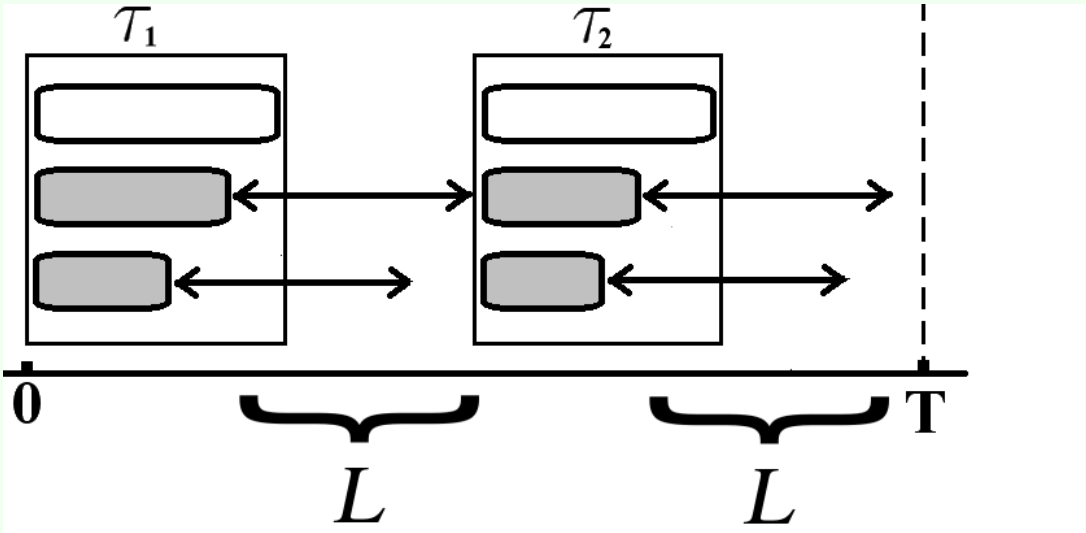
# Computing the stability radius

A special case: if $\boldsymbol{T}$ is large enough.



# New formulation

A special case: if $\boldsymbol{T}$ is large enough then the solution is always feasible and $\boldsymbol{\rho = L\tilde{n}}$.



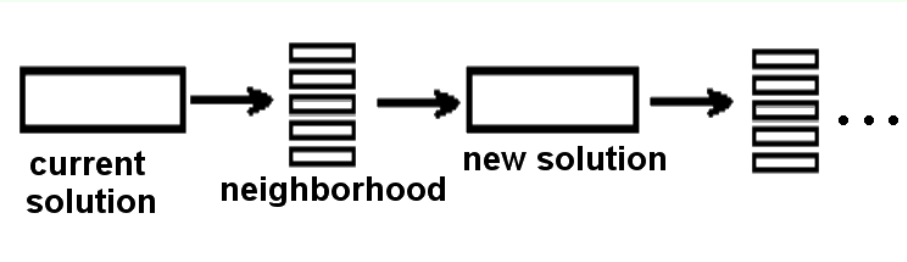
# Computing the stability radius

Computing the stability radius for one workstation

1. Sort blocks by increase of their save times
2. Let $\boldsymbol{\rho := 0}$, $\boldsymbol{R := T - \sum_{k=1}^{K} \tau_k}$.
3. For $\boldsymbol{k} = 1$ to $\boldsymbol{K}$ do:
   2.1. Let $\boldsymbol{a_k := L - \Delta_k}$.
   2.2. If $\boldsymbol{a_k < 0}$ then
   stop and **return** $\boldsymbol{L\tilde{n}}$.
   2.3. If $\boldsymbol{R < a_k}$ then
   set $\boldsymbol{a_k := R,\ \ \rho := \rho + \Delta_k + a_k}$, stop and **return** $\boldsymbol{\rho}$.
   Else
   $\boldsymbol{R := R - a_k}$, $\boldsymbol{\rho := \rho + \Delta_k + a_k}$.
4. **Return** $\boldsymbol{L\tilde{n}}$.

*Theorem.* The algorithm correctly finds the stability raduis.

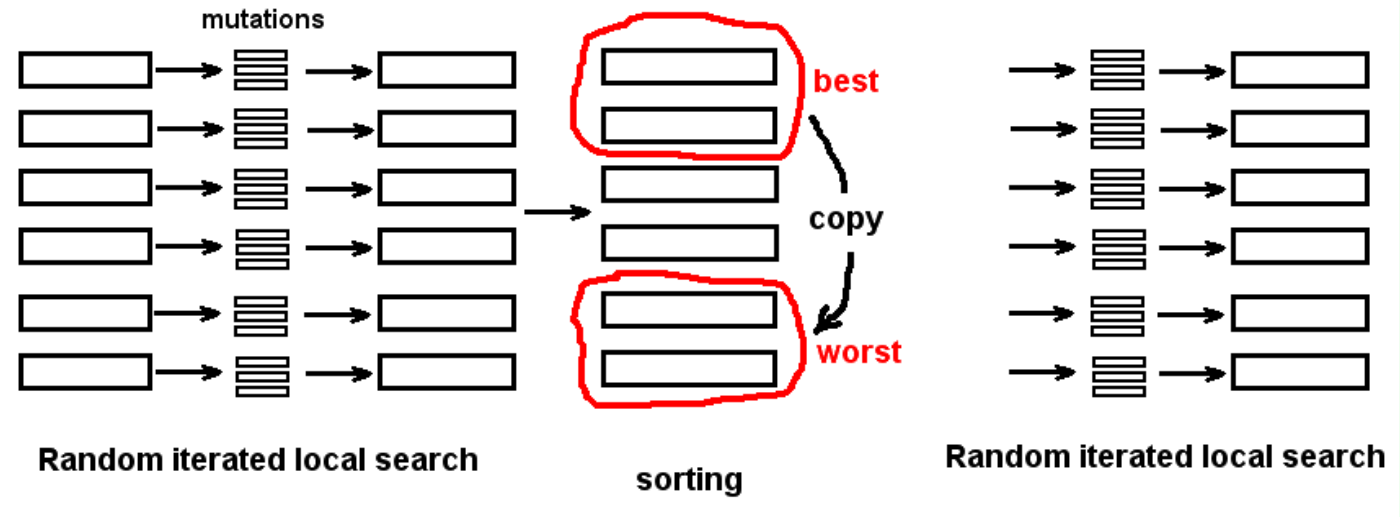
# Random Iterated Local Search




On each iteration:

1. Apply small random changes (*mutation*) many times in parallel.
2. Select the best output and make it the new currect solution.
3. On certain iterations, a *shaking* procedure is applied, which consists in several mutations in a sequence.

# Hybrid "Go with the Winners" algorithm




Mutations and evaluation of obtained solutions are done in parallel on a Graphics Processing Unit (GPU)

Aldous, D., Vazirani, U.: "Go with the winners" algorithms. In: Proc. 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, pp 492–501. IEEE (1994).

Borisovsky, P.: A parallel "Go with the winners" algorithm for some scheduling problems. Journal of Applied and Industrial Mathematics (in press)

# Computational Experiments

GPU Tesla V100 (1530 MHz, 5120 CUDA cores)
Test instances from (Pirogov et al., 2021).

| Series S1 $n = 25, m = 5$ $L = 20$ | optimize unlimited | | optimize limited | |
|---|---|---|---|---|
| | $\rho_U^{best}$ | $\rho_L^{avg}$ | $\rho_U^{avg}$ | $\rho_L^{best}$ |
| S1.0 | 30.4 | 30.4 | 20.59 | 260 |
| S1.1 | 27.3 | 27.3 | 16.52 | 38.6 |
| S1.2 | 29.0 | 29.0 | 21.9 | 30.5 |
| S1.3 | 34.5 | 34.5 | 20.99 | 260 |
| S1.4 | 28.5 | 28.5 | 14.26 | 36.7 |
| S1.5 | 32.3 | 32.3 | 21.54 | 260 |
| S1.6 | 28.5 | 28.5 | 20.17 | 35.4 |
| S1.7 | 38.5 | 38.5 | 20.62 | 260 |
| S1.8 | 32.4 | 32.4 | 21.7 | 260 |
| S1.9 | 24.4 | 24.4 | 23.87 | 24.4 |

In the unlimited case. Metaheuristic: 10 times by 1 second
Gurobi: 10 minutes running time limit.

# Conclusions

- A new robust transfer line balancing problem with limited exctra processing times is considered.
- A calculation of the stability radius is provided and validated.
- A parallel metaheuristic for maximization of the stability radius is developed.
- It would be worthwhile to extend this approach to the case, where uncertain tasks have different time limits.

Thank you for your attention!