# Graphical Abstract

**Actor-based Large Neighborhood Search**

Christian Brunbjerg Jespersen

# Highlights

**Actor-based Large Neighborhood Search**

Christian Brunbjerg Jespersen

- How to allow direct and real-time integration into an optimization process?
- How to perform optimmization in a real-time changing parameter space?

# Actor-based Large Neighborhood Search

Christian Brunbjerg Jespersen

[a], , , , ,

**Abstract**

Abstract text.

*Keywords:* Large Neighborhood Search, Actor Framework — Real-time Optimization

## 1. Introduction

Many problems that face the operation research field are hard to solve due to the need of tight integration within companies that (meta)heuristics and models are usually tested with. To solve operation research problems that are highly operational in nature there is a need for metaheuristics that are more responsive, that work with the user, and that allows for tight integration into user work-flows Doe and Roe. Operations in are often characterized by extensive amounts negotiation and feedback on proposed solutions to general operations research problems (!!!). This often lead to a situation where operation research methods are not used to solve the problem that their underlying model proposes to solve and instead end up providing suggestions instead of a complete solution (!!!). Operations in are often characterized by extensive amounts negotiation and feedback on proposed solutions to general operations research problems (!!!). This often lead to a situation where operation research methods are not used to solve the problem that their underlying model proposes to solve and instead end up providing suggestions instead of a complete solution (!!!). This paper proposes a solution method that will allow for real-time optimization based on user-interaction (effective altering the parameter space) and real-time connection to data sources, such to a database connections. To illustrate the solution method a variant of the knapsack problem has been chosen, specifically the multi-compartment multi-knapsack problem on a large dataset. Furthermore the large neighborhood

search metaheuristic has been chosen due to its properties of being able to work with and fix infeasible solutions.

The paper is divided into four different sections. Section 2 explains the model and method in detail that form the fundation of the paper. Section **??** shows that results coming from the implemented system where the system will be affected by simulated user-interaction. Section **??** will discuss the implications of the research and possible future research directions.

## 2. Solution Method

### *2.1. The Multi-compartment Multi-knapsack Problem with capacity penalties*

Here a variant of the knapsack problem is chosen as the problem to apply actor-based large neighborhood search on. It has been chosen due to its simplicity while also being sufficiently computationally hard to be able to illustrate the rationale behind the solution approach. The model is comprised of five different sets. Here: $K$ is the number of knapsacks; $I$ is the number of items; $C$ is the number of compartments; $Q$ is a set that defines which items should be excluded from a specific knapsack; $P$ is an inclusion set that defines the items which should be included in each knapsack. The model introduces four different parameters. Here: $v_{ik}$ is the value associated with item i in knapsack k; $d$ is the penalty paid for exceeding the compartment capacity; $w_{ic}$ is the capacity requirement for item i in compartment c; $cap_{kc}$ is the total amount of capacity available in knapsack k for compartment c. The decision-variables are: $x_{ik}$, which is a binary decision variable equal to one if item i is in knapsack k and zero otherwise; $p_{kc}$ is non-negative decision variable equal to the amount of excess capacity above the $c_{kc}$ in knapsack k for compartment c.

$$\text{Min} \quad \sum_{i=1}^{I}\sum_{k=1}^{K} v_{ik} \cdot x_{ik} - \sum_{k=1}^{K}\sum_{c=1}^{C} d \cdot p_{kc} \tag{1}$$

subject to:

$$\sum_{i=1}^{I} w_{ic} \cdot x_{ik} \leq \ cap_{kc} + p_{kc} \qquad \forall k \in K, \forall c \in C \tag{2}$$

$$\sum_{i=1}^{I} x_{ik} \leq \ 1 \qquad \forall k \in K \tag{3}$$

$$x_{ik} = 0 \qquad \forall (i,k) \in Q \tag{4}$$

$$x_{ik} = 1 \qquad \forall (i,k) \in P \tag{5}$$

$$x_{ik} \in \{0,1\} \qquad \forall i \in I, \forall k \in K \tag{6}$$

$$p_{kc} \in \mathbb{R}^{+} \qquad \forall k \in K, \forall c \in C \tag{7}$$

Here a series of knapsacks

*2.2. Actor-based Large Neighborhood Search*

**Algorithm 1** Actor-based Large Neighborhood Search

```
1: Input queue = message queue
2: Input P = problem instance
3: Input x = initial solution
4: x^b = x
5: repeat
6:     if !queue.is_empty then
7:         m = queue.pop()
8:         x.update(m)
9:         x^b.update(m)
10:        P.update(m)
11:    end if
12:    x^t = r(d(x))
13:    if accept(x^t, x) then
14:        x = x^t
15:    end if
16:    if c(x^t) < c(x^b) then
17:        x^b = x^t
18:        queue(x^b)
19:    end if return x^b
```

## References

Doe, J., Roe, J., . An example article. Journal of Examples 10, 210–215.