

# 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 optimization in a real-time changing parameter space?

# Actor-based Large Neighborhood Search

Christian Brunbjerg Jespersen

<sup>a</sup>, , , , ,

---

## Abstract

Abstract text.

*Keywords:* Large Neighborhood Search, Actor Framework, Real-time Optimization, human-centered computing, Interactive Systems and Tools, Decision Support Systems, Interactive Optimization

---

## 1. Introduction

Many problems that face the operation research field are hard to solve due to the need of tight integration tacit knowledge found in the minds of the the decision makers that have to make decision based on solutions provided by operation research methods such as metaheuristics and models. To solve operation research problems that are highly operational in nature there is a need for metaheuristics that are: 1. responsive, 2. able to be assimilated into the decision makers workflow, and 3. allows for integration into dynamic data source such as traditional databases (Meignan et al., 2015). Operational aspects of operation research, as opposed to strategic and tactical aspects, are often characterized by 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 provides intial suggestions instead of a complete solution Meignan et al. (2015) . Operations are often characterized by extensive amounts negotiation and feedback on proposed "solutions" between relevant stakeholders. 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.

This paper proposes a solution method that will allow for real-time optimization based on user-interaction (effectively altering the parameter space)

and real-time connection to data sources, such to a database connections. The proposed solution method will be based on 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 foundation 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.

### *1.1. 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  $c$ .

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

subject to:

$$\sum_{i=1}^I w_{ic} \cdot x_{ik} \leq \text{cap}_{kc} + p_{kc} \quad \forall k \in K, \forall c \in C \quad (2)$$

$$\sum_{i=1}^I x_{ik} \leq 1 \quad \forall k \in K \quad (3)$$

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

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

$$x_{ik} \in \{0, 1\} \quad \forall i \in I, \forall k \in K \quad (6)$$

$$p_{kc} \in \mathbb{R}^+ \quad \forall k \in K, \forall c \in C \quad (7)$$

In this paper we will specifically look at the effects of changing Q, P, cap, and v in real-time and illustrate the effects that this approach has on the solution and the objective value.

## 2. Solution Method

### 2.1. Large Neighborhood Search

### 2.2. Actor-based Large Neighborhood Search

A problem that is subject to user-interaction and requires real-time feedback you need an optimization approach that is able to repair infeasible solutions and while also converging quickly. For this the large neighborhood search metaheuristic has been shown satisfy these requirements in the literature ?.

The LNS metaheuristic is in its most basic form implicitly defined for static problems meaning that the parameters that make up the problem

instance is not subject to change after the algorithm has been started. To make the LNS able adapt to incoming information in real-time a message system have been implemented on top of the existing framework.

This extension is shown in algorithm 1.

---

**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:      $x^b = x$ 
18:     queue( $x^b$ )
19:   end if return  $x^b$ 

```

---

The basic LNS setup have here been extended with a ‘message queue’. This message queue will be read from on every iteration of the LNSs main iteration loop. And here we should notice that the incoming message are able to change both the solutoin but also the problem instance itself. Here we see one of the defining features of the LNS metaheuristic in play, that due to its inherrent property of being able to optimize a solution that have become infeasible which is something that is very likely to happen when you change the parameter of the problem instance itself.

Another less obvious property the message queue allows is for the algorithm to run indefinitely and instead of restarting the algorithm you instead pass messages to it to allow it be adjust both the solution space and the parameter space. This property avoid the issue of time consuming initial

convergence as the algorithm will be found in an optimal state when the solution is perturbed.

### **3. Results**

This results section will show how the proposed solution method will react to incoming inputs into the

### **4. Discussion**

(MICROSERVICE SOURCE)

### **References**

Meignan, D., Knust, S., Frayret, J.M., Pesant, G., Gaud, N., 2015. A Review and Taxonomy of Interactive Optimization Methods in Operations Research. ACM Transactions on Interactive Intelligent Systems 5, 1–43. URL: <https://dl.acm.org/doi/10.1145/2808234>, doi:10.1145/2808234.