Organisations (de-)centralised to a greater or lesser degree for allocating cities in two Multiple Travelling Salesmen Problems

Thierry Moyaux

Univ Lyon, INSA Lyon, Univ Jean Monnet Saint-Étienne, Université Claude Bernard Lyon 1, Univ Lyon 2, DISP-UR 4570, Villeurbanne G-69621, France
Tel.: (+33) 4 72 43 75 37

Abstract

Decisions in organisations may be made either by a Central Authority (CA), e.g., in a hierarchy, or by the agents in a decentralised way, e.g., in a heterarchy. Since both kinds of organisations have their advantages (e.g., optimality for centralised organisations and reactivity for decentralised ones), our goal is ultimately to understand when and how to use each of them. Our previous work proposed a variant of the Multiple Travelling Salesmen Problem, which we now call MTSP$_s$. We use the subscript “s” to refer to salesmen’s selfishness when they minimise their individual route length. If, on the contrary, they are assumed to be benevolent, we add subscript “b” and thus the term MTSP$_b$ to refer to the traditional MTSP in which the salesmen minimise the total route length. This article shows how to obtain such benevolent agents by slightly modifying selfish agents. We can then compare organisations which are (de-)centralised to a greater or lesser degree, which enables us to carry out the allocation of cities in the MTSP$_b$. The first experiment shows that the relative efficiency (ranking) of the organisations differs between MTSP$_b$ and MTSP$_s$. Since reactivity fosters decentralisation, the second experiment gradually reduces the time taken for it to impact this ranking. Both experiments show that pure centralisation is either the best or the worst option, and that the zone between the two situations is very narrow.

Keywords: Decision making, Centralised decision, Decentralised decision, MTSP

1. Introduction

The structure of an organisation may be (de-)centralised to a greater or lesser degree. We call “organisation” the structure of this organisation, and “mechanism” an instance of organisation. Our goal is ultimately to establish which structure is best suited to a situation. To this end, this article compares organisations that are (de-)centralised to a greater or lesser degree, to solve the MTSP (Multiple Travelling Salesmen Problem), i.e., the Vehicle Routing Problem without capacity constraints [13, 1]. This problem consists of two connected subproblems: allocation and routing. This article studies the (de-)centralisation of the former. In this context, our research question may be stated as: when is it more efficient to let salespersons/agents concurrently solve many small Travelling Salesman Problems (TSP, i.e., the routing subproblem alone), rather than letting a Central Authority (CA) solve the single large Multiple Travelling Salesmen Problem (MTSP, i.e., both allocation and routing problems together)? We call the centralised organisation Centr. The others are decentralised organisations and we will see that they have various levels of decentralisation.

This research question is important because optimising the operations in a bad organisation may not be as efficient as improving the organisation itself. For example, (de-)centralising the maintenance of buildings in a university seems to be a completely different question from that of (de-)centralising the maintenance of its computers, as the required reactivity of these two types of maintenance is very different. Repairing lights, doors, toilets, roofs, and so forth can wait for a few hours without completely bringing to a halt teaching, research and administration, whereas a failure of computers hardware or software disturbs such operations significantly. When an organisation is in place, it is very difficult to change it because this would impact: (i) technologies supporting decisions, i.e., the information system in terms of structure, technologies (e.g., need for big servers and/or smaller devices such as laptops and smartphones), tools, etc.; (ii) resistance to change (power – the Central Authority (CA) in a centralised organisation will never give up their power to switch to a decentralised organisation – and habits); (iii) loss of experience (processes and tools previously optimised will be replaced by new unoptimised ones); (iv) etcetera.

This question has been addressed by various communities, as our literature review in Economics and Computer Science literature review attests [8]. In particular, this review points to a comparison of four organisations to solve a resource allocation problem for production and transportation by [3] which, to our knowledge, is the only quantified comparison of as many as four organisations. More recently, [2] have reviewed the literature on Holonic Control Architectures and concluded that additional research on the coupling of centralised and decentralised mechanisms in a same organisation should address the following three problems: (i) the estimation of future performance because a disturbance may have either no impact or a huge impact on the system; (ii) the design of switching indicators which use the estimation in i to decide if and when to
switching from a centralised to a decentralised mechanism; and (iii) the strategy for switching between both mechanisms seamlessly without stopping the system. Our work addresses the first problem. It moreover resembles that of [14], but in the area of transportation rather than production scheduling. More precisely, these authors review traditional (centralised) scheduling in order to propose future research direction for “smart distributed scheduling” in Industry 4.0. In fact, [4] identify “Autonomous and decentralised decision support systems” as one of the ten key enablers of Industry 4.0. As noted above, some of our organisations are not purely (de-)centralised, which is also investigated in the area of manufacturing systems by [10], among others. Concerning MTSP, [5] propose a very interesting decentralised mechanism – different from ours – for the organisation which we call P2P, that is, Peer to Peer with benevolent agents.

In this article we propose to take a centralised problem that has been thoroughly investigated by operational research and to study when and how a combination of operational research and multiagent techniques may decentralise it efficiently. Since operational research is able to guarantee the optimality of a solution when enough time is available, we set a time limit in order to quantify the reactivity of organisations. This is consistent with the intuition that decentralisation is more efficient when reactivity constraints the time available to make decisions. From a more technical perspective, we explore how agents/salespersons may solve the MTSP in organisations with a greater or lesser degree of (de-)centralisation. We call the traditional version of this problem MTSP\textsubscript{p}, for MTSP with benevolent agents, in the sense that they minimise the sum of the individual route lengths of all the agents, that is, the total route length. In contrast, our previous article [7] proposes MTSP\textsubscript{s} in which the agents are selfish as they want to minimise their individual route length only. Our contributions may be summarised as follows:

1. **Quantification of the efficiency of organisations that are (de-)centralised to a greater or lesser degree:** Our experimental results show the performance of each mechanism when it solves a set of instances compared to the performance of the centralised organisation when it solves the same instances. More precisely, we show the median and ninth decile of the ratios of the total route length found by a mechanism on these instances divided by the total route length found by “Centr\textsubscript{p}”. As noted above, we set a limit to the computation time such that this ratio may be lower than 1 when Centr\textsubscript{p} does not have enough time to find the best allocation and routings.

2. **Many small problems vs. a single large problem:** The previous contribution is equivalent to the identification of when a CA or the salesmen solve the routing subproblem, (ii) have zero or one CA (who may be either coercive or a facilitator), (iii) make either the CA or the salesmen solve the routing subproblem, (iv) etc. The organisations in Figure 1 may be briefly described as:

   - **Centr\textsubscript{p}**: The Central Authority (CA) is also an agent, but we never need to refer to her this way. We use “he” to designate a salesman who may also be called agent, and “she” for the Central Authority (CA), CA is also an agent, but we never need to refer to her this way. We call “total route length” the sum of the “individual route lengths” of all agents/salesmen. Figure 1 shows the organisation considered in this article. We call mechanism an instance of an organisation, e.g., organisation Cluster\textsubscript{p} will be instantiated below as Mechanisms Cluster\textsubscript{p} and Cluster\textsubscript{s}. We can also see in this figure that our organisations may (i) require a single or multiple rounds of interactions, (ii) have zero or one CA (who may be either coercive or a facilitator), (iii) make either the CA or the salesmen solve the routing subproblem, (iv) etc. The organisations in Figure 1 may be briefly described as:

3. **Same models for allocation and routing:** In order to focus on the impact of (de-)centralisation only, we try to keep all other things equal. In particular, we propose to minimise the impact of implementation by (i) taking only the computation time of the solver of MILP (Mixed Integer Linear Program) CPLEX into account, (ii) proposing a same allocation MILP for all our decentralised mechanisms, and (iii) relying on close MILP formulations of MTSP and TSP.

4. **Same structure of state charts for selfish and benevolent agents:** In our previous article, the agents were selfish, whereas this article assumes they are benevolent. It is interesting to note that the structure of the state charts of all our mechanisms remains the same. Only the action carried out in some of the states needs to be modified and more information is exchanged between the agents. Therefore, we do not show these state charts again instead, we show our mechanisms through their sequence diagram, which is an equivalent way of presenting them. Moreover, two footnotes suggest other mechanisms for benevolent agents with the same structure of state charts.

5. **Experimental conclusions drawn from two variants of MTSP:** Not only do we show the results for MTSP\textsubscript{p}, we also summarise the results for MTSP\textsubscript{s} reported in our previous article, and note that they are qualitatively different. They are therefore presented as different rankings. The reader does not need to read our previous article as we summarise it whenever we cite it.

The outline of this paper is as follows. Section 2 describes all our mechanisms. Section 3 details two experiments. Section 4 concludes.

2. **Our allocation mechanisms to solve the traditional MTSP\textsubscript{p}**

This section first introduces definitions and hypotheses before presenting our mechanisms.

2.1. Definitions

MTSP\textsubscript{p} and MTSP\textsubscript{s} are the optimisation of two subproblems, namely, the allocation of n cities to m salesmen, and the routing of each salesman such that he visits all the cities allocated to him. We use “he” to designate a salesman who may also be called agent, and “she” for the Central Authority (CA), CA is also an agent, but we never need to refer to her this way. We call “total route length” the sum of the “individual route lengths” of all agents/salesmen. Figure 1 shows the organisations considered in this article. We call mechanism an instance of an organisation, e.g., organisation Cluster\textsubscript{p} will be instantiated below as Mechanisms Cluster\textsubscript{p} and Cluster\textsubscript{s}. We can also see in this figure that our organisations may (i) require a single or multiple rounds of interactions, (ii) have zero or one CA (who may be either coercive or a facilitator), (iii) make either the CA or the salesmen solve the routing subproblem, (iv) etc. The organisations in Figure 1 may be briefly described as:

---

Please refer to [7] for a presentation of the state charts of our mechanisms. In addition, we will add our AnyLogic model and the outcomes of the experiments to [https://github.com/disp-lab/centr_vs_decentr](https://github.com/disp-lab/centr_vs_decentr) after this article is accepted for publication.
Exchange of messages
Allocation problem
Routing problem
Operator (i.e., salesman)
Central authorithy (CA)

NoRealloc

Coercion level
Powerless central authority
(Coercive central authority)
(Centralised)

1−n interactions
1−1 interactions
1−1 interactions
1−n interactions
1−1 interactions
1−n interactions
1−n interactions
1−1 interactions
1−n interactions

No central authority
(Single round of interactions)

Central authority (CA)
Allocation problem
Routing problem
Exchange of messages

Figure 1: Overview of the allocations mechanisms.
NoRealloc is a benchmark showing the worst/longest total route length by ignoring the allocation subproblem.

Centr finds the best/shortest total route length (when the time limit is sufficient) in a single round by letting CA jointly optimise allocation and routings.

CNPb is inspired by the Contract Net Protocol proposed by [12]. In each round, one of the salesmen plays the role of the auctioneer of an auction to exchange cities.

Auction also operates by rounds in which the CA plays the role of the auctioneer.

P2Pb also runs by rounds but only involves two agents who try to exchange cities.

Cluster works in a single round in which all salesmen send their initial endowment of cities to CA, then CA creates clusters of these cities, and the salesmen finally solve a TSP with the cities sent by CA. This organisation has two variants called Cluster and ClusterS, depending on the formulation of the clustering MILP used by CA.

We name “cost” of a city the additional route length necessary to visit this city, that is, the difference between the lengths of the routes for visiting and not visiting this city. During an exchange of cities called initial endowment.

We make an important assumption about the durations taken into account in both our experiments. We choose to consider only the computation time of the solver CPLEX. This is a solution to the problem of comparing, for example, (i) a piece of software written in Java within a month by one person and (ii) a professional library such as CPLEX written in C and improved for decades by a team of developers. In fact, what we want to study is the impact of (de-)centralisation and we thus try to keep everything else as identical as possible. This assumption incurs consequences, such as the impossibility to use techniques such as k-means for clustering, the belief–desire–intention software model to represent agents, the neglecting of the travelling time of messages, etc.

Moreover, we try to make CPLEX solve problems as similar as possible in the various mechanisms. First, we will see that the formulation of TSP in Equations [16] MTSP in Equations [7] and a modified TSP finding the city with the highest cost in Equations [15] are very close. Next, we introduce the allocation problem in Equations [25] NoRealloc such that it can be shared by all our decentralised mechanisms.

Finally, we assume that the agents start with an allocation of cities called initial endowment.

2.3. Mechanism NoRealloc

NoRealloc is not really a mechanism but a benchmark showing the upper (worst) bound of the total route length when the allocation subproblem is ignored. No exchanges of messages or cities take place and each salesman solves only the TSP in Equations [16] with the cities in his initial endowment.

\[
\begin{align*}
\min & \quad \sum_{i=0}^{N-1} \sum_{j=0, j \neq i}^{N-1} d_{ij} x_{ij} \\
\text{s.t.} & \quad \sum_{j=0, j \neq i}^{N-1} x_{ij} = 1 \quad 0 \leq i < N \\
& \quad \sum_{i=0, i \neq j}^{N-1} x_{ij} = 1 \quad 0 \leq j < N \\
& \quad p_i - p_j + N x_{ij} \leq N - 1 \quad 1 \leq i \neq j < N \\
& \quad p_i \in \mathbb{R}^+ \quad 1 \leq i < N \\
& \quad x_{ij} \in \{0, 1\} \quad (i, j) \in \{0, 1, \ldots, N - 1\}
\end{align*}
\]

In this MILP, the binary decision variable \( x_{ij} \) in Equation (6) equals 1 only if the considered salesman goes from City \( i \) to City \( j \), and zero otherwise. Equation (1) minimises the length of this salesman’s individual route, which means that the distance \( d_{ij} \) between Cities \( i \) and \( j \) is added up only when \( x_{ij} = 1 \). This sum is up to \( N \neq n \) which is the number of cities currently allocated to the considered salesman (it is not necessary to call it \( N_{a} \) as it is a variable local to salesman \( a \)). Equation (2) ensures that City \( i \) (respectively, \( j \)) is left (respectively, entered) exactly once. The constraint in Equation (3) eliminates sub-routes by the node potentials method [6] which uses the real decision variable \( p_i \) in Equation (4) to count the number of cities visited by the considered salesman before he visits City \( i \).

2.4. Mechanism Centr

Basically, Centr operates in a single round in which CA solves a well-known MILP formulation of MTSPb and the salesmen make no decisions. More precisely, (i) the \( m \) salesmen send a message with their initial endowment, (ii) CA jointly solves both allocation and routing subproblems in Equations [7] and (iii) CA sends to each of the \( m \) salesmen a message with
of modelling, implementation or anything else. Which is a desired feature if we wish to study the impact of Equations 1-6 and MTSP in Equations 7-14 look very similar, respectively, enters) the depot. As highlighted above, MTSP in Figure 2: Interactions in every round of P2Pb. namely City zero which is allocated to all salesmen. Equation 8 (respectively, 11) is similar to Equation 2 (respectively, 3) except that the right-hand side does not read 1, but Decision Variable kepti; according to Equation 18 kepti = 1 for all cities except the one to be proposed to h and which we call cG. Finally, g memorises in the matrix of Booleans propCities[h][i][20] whether City i has been proposed to this host h in a previous round or not, and Equation 20 ensures that cG cannot be City i, i.e., City i cannot be proposed to agent h again. Next, g calculates dG as the difference between his current route length and the value of the objective function in Equation 15.

3. If the interaction has not been stopped by Message 2, i.e., g did not reply 0, then h chooses a City cG by also solving the modified TSP in Equations [15][24] (of course, the roles of guest and host need to be swapped in the above explanation of this MILP). Then, h sends cG and the potential reductions of cost dG, dG and dG,cG to g. An example below illustrates the calculation of these three cost reductions.

4. Guest g finds the best allocation for all salesmen and either (Case 1) keeps both cG and cG by sending 0 to h, or (Case 2) keeps cG by sending cG (the two cities are exchanged), or (Case 3) keeps cG and returns cG (no exchange of cities), or (Case 4) keeps nothing and sends both cG and cG. Since we only take the computation time of CPLEX into account, we need to write the selection of the modified TSP in Equations [15][24] in order to identify cG.

\[
\min \sum_{i=0}^{n-1} \sum_{j=0, j\neq i}^{n-1} d_{ij} x_{ij} \quad (7)
\]

\[
\sum_{j=0}^{n-1} x_{0j} = m \quad (8)
\]

\[
\sum_{j=0, j\neq i}^{n-1} x_{ij} = 1 \quad 1 \leq i < n \quad (9)
\]

\[
\sum_{j=0, j\neq i}^{n-1} x_{ij} = 1 \quad 1 \leq j < n \quad (10)
\]

\[
\sum_{i=0}^{n-1} x_{0i} = m \quad (11)
\]

\[
p_i - p_j + (n - 1) - x_{ij} \leq n - 2 \quad 1 \leq i \neq j < n \quad (12)
\]

\[
p_i \in \mathbb{R}^+ \quad 1 \leq i < n \quad (13)
\]

\[
x_{ij} \in \{0, 1\} \quad 0 \leq i, j < n \quad (14)
\]

\[
\sum_{i=0}^{n-1} \sum_{j=0, j\neq i}^{n-1} c_{ij} x_{ij} \quad (15)
\]

s.t.

\[
\sum_{j=0, j\neq i}^{n-1} x_{ij} = \text{kept}_i \quad 0 \leq i < N \quad (16)
\]

\[
\sum_{i=0}^{n-1} x_{ij} = \text{kept}_j \quad 0 \leq j < N \quad (17)
\]

\[
\sum_{i=0}^{n-1} \text{kept}_i = N - 2 \quad 0 \leq i < N \quad (18)
\]

\[
p_i - p_j + N \cdot x_{ij} \leq N - 1 \quad 1 \leq i \neq j < N \quad (19)
\]

\[
\text{kept}_i = 1 \quad \text{propCities}[h][i][20] = \text{true} \quad (21)
\]

\[
\text{kept}_i \in \{0, 1\} \quad i \in \{1, \ldots, N - 1\} \quad (22)
\]

\[
p_i \in \mathbb{R}^+ \quad 1 \leq i < N \quad (23)
\]

\[
x_{ij} \in \{0, 1\} \quad (i, j) \in \{0, 1, \ldots, N - 1\} \quad (24)
\]

\[
G \quad \text{Number of guests; in P2Pb, } G = 1 \quad (and \ m - G = 1 \text{ other salesman is the host}).
\]

\[
A \quad \text{Set of participating agents/salesmen; in P2Pb, } A = \{h, g\}.
\]
• \( C \): Set of cities that are proposed in this round of exchange; In \( P2P_b \), \( C = \{ c^h, c^g \} \).

• \( \bar{C} \): Set of bundles of one or two cities that are proposed in this round of exchange. \( \bar{C} \) is derived from \( C \). In \( P2P_b \), \( \bar{C} = C \cup \{ [c^h, c^g], [c^h, c^h], [c^g, c^g] \} \), viz., three bundles (two single-city bundles and one two-city bundle).

• \( C^a \): Set of cities currently allocated to Agent \( a \); in any round of \( P2P_b \), \( c^h \in C^h \) and \( c^g \in C^g \), i.e., the city proposed for an exchange by an agent is allocated to this agent in the considered round.

• \( |X| \): Number of elements in set \( X \); for example, in any round of any decentralised mechanism, \( |C^a| \geq 2 \) because all agents \( a \) are allocated at least the depot plus one city in order to comply with MTSPb in Equations 7 and 14. As another example, in any round of \( P2P_b \), \( |C| = 2 \) because exactly two agents propose one city each.

• \( d_{\bar{c}}^0 \): Cost (additional distance) for Agent \( a \) to visit Bundle \( \bar{c} \in C \). As noted above, for any two cities \( c1 \) and \( c2 \), \( d_{c1,c2}^0 = \delta_{c1,c2}^0 \), but an example below will show that \( d_{c1,c2}^0 \) may be either smaller, equal or larger than \( d_{c1}^0 + d_{c2}^0 \), i.e., this cost does not follow the triangle inequality. In \( P2P_b \), \( [d_{\bar{c}}^0] = [d_{c1}^0 \quad d_{c2}^0 \quad d_{c1,c2}^0] \).

This example shows that \([\cdot]\) is a matrix, e.g., \( d_{c1}^0 \) points to one specific value in Matrix \([d_{\bar{c}}^0]\).

• \( w_{\bar{c}}^0 \): Weight associated with Bundle \( \bar{c} \), i.e., \( w_{\bar{c}}^0 \in C \). In \( P2P_b \), \( [w_{\bar{c}}^0] = [1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1] \).

Again, Matrix \([w_{\bar{c}}^0]\) contains all such values.

• \( v_{\bar{c}} \): This binary decision variable equals one only if Bundle \( \bar{c} \) is allocated to agent \( a \), and zero otherwise.

• \( x_{\bar{c}} \): This binary decision variable equals one only if Bundle \( \bar{c} \) is allocated to agent \( a \), and zero otherwise.

Guest \( g \) uses CPLEX to solve the allocation MILP in Equations 25-29:

\[
\begin{align*}
\min & \quad \sum_{\bar{c}\in\bar{C}} \sum_{c\in C} x_{\bar{c}}^c d_{\bar{c}}^0 \\
\text{s.t.} & \quad \sum_{c_1 \in C} x_{\bar{c}}^c , w_{\bar{c}}^0 = 1 \quad \forall c \in C \\
& \quad \sum_{c \in C} x_{\bar{c}}^c , v_{\bar{c}} \leq 1 \quad \forall a \in A \\
& \quad \sum_{c \in C} x_{\bar{c}}^c , v_{\bar{c}} \geq 3 - |C^a| \quad \forall a \in A \\
& \quad x_{\bar{c}}^c \in \{0, 1\} \quad \forall a \in A, \bar{c} \in \bar{C}
\end{align*}
\]

The objective function in Equation 25 minimises the sum of the costs of the salesmen involved in this round of exchange, e.g., \( h \) and \( g \) in \( P2P_b \). This minimisation corresponds to a reduction of the total route length travelled by all agents as they are benevolent. Equation 26 ensures that Bundle(s) \( \bar{c} \) is/are selected such that each City \( c \) is visited exactly once. Equation 27 prevents the allocation of a pair of cities instead of the two-city bundle containing them, which may occur otherwise because costs \( d_{\bar{c}}^0 \) do not follow the triangle inequality as illustrated in the example in the next paragraph. The constraint in Equation 28 ensures that all agents will be allocated at least two cities (the depot plus another city): the left-hand side counts the number of cities in allocation \( x_{\bar{c}}^c \) and the right-hand side is the minimal number needed for this agent to keep at least two cities.

**Example**

We illustrate how the allocation is performed in \( P2P_b \) on an example showing that Matrix \([d_{\bar{c}}^0]\) does not follow the triangle inequality, and why the constraints in Equation 27 are therefore required. These constraints forbid the allocation of two single-city bundles to the same agent because the corresponding sum of costs would be different from the cost of allocating the bundle of these two cities. This example is shown in Figure 3 (a) and (b). Guest \( g \) owns the depot \( c_0 \) located at \( (0, 0) \) and Cities \( c_1 \) located at \( (50, 200) \), \( c_3 \) at \( (50, 150) \) and \( c_5 \) at \( (200, 150) \). Host \( h \) is initially allocated the depot \( c_0 \), as well as Cities \( c_2 \) at \( (100, 50) \) and \( c_4 \) at \( (0, 200) \). In the first round of interaction, \( h \) tries to get rid of \( c^h = c_1 \), and \( g \) of \( c^h = c_5 \). \( g \) solves the allocation subproblem in Equations 25-29 with

\[
[d_{\bar{c}}^0] = \begin{bmatrix}
    d_{c1}^0 & d_{c2}^0 & d_{c1,c2}^0 \\
    d_{c3}^0 & d_{c4}^0 & d_{c3,c4}^0 \\
    d_{c5}^0 & d_{c6}^0 & d_{c5,c6}^0
  \end{bmatrix} = \begin{bmatrix}
    268.47 & 43.84 \\
    279.61 & 201.95 \\
    435.77 & 272.66
  \end{bmatrix}
\]

Let us detail the computation of the right-hand column of this matrix:

- \( D^h \) is used in the next three bullet points to denote the optimal distance to visit all the cities allocated to \( g \) excluding both \( c^h \) and \( c^g \).
- \( d_{\bar{c}}^0 \) is the difference between (i) the optimal distance to visit all the cities allocated to \( g \) including \( c^h \) and excluding \( c^g \), and (ii) \( D^h \).
\begin{itemize}
\item $d_G^h$ is the difference between (i) the optimal distance to visit all the cities allocated to $g$ excluding $c^h$ and including $c^e$, and (ii) $D_G$.

\item $d_G^{c^h,c^e}$ is the difference between (ii) the optimal distance to visit all the cities allocated to $g$ including both $c^h$ and $c^e$, and (ii) $D_G$.
\end{itemize}

Solving the TSP in Equations 11 gives the following four optimal routes:

\begin{itemize}
\item $D_G^e$ is the length of the route $c_0 \rightarrow c_3 \rightarrow c_1 \rightarrow c_0$, hence $D_G = \sqrt{(0 - 50)^2 + (0 - 150)^2} + \sqrt{(50 - 50)^2 + (150 - 200)^2} + \sqrt{(50 - 0)^2 + (200 - 0)^2} = 414.27$.

\item $d_G^c$ is the length of the route $c_0 \rightarrow c_4 \rightarrow c_1 \rightarrow c_3 \rightarrow c_0$ minus $D_G$, hence $d_G^c = 458.11 - 414.27 = 43.84$.

\item $d_G^{c^h}$ is the length of the route $c_0 \rightarrow c_5 \rightarrow c_1 \rightarrow c_3 \rightarrow c_0$ minus $D_G$, hence $d_G^{c^h} = 616.23 - 414.27 = 201.95$.

\item $d_G^{c^h,c^e}$ is the length of the route $c_0 \rightarrow c_5 \rightarrow c_1 \rightarrow c_4 \rightarrow c_3 \rightarrow c_0$ minus $D_G$, hence $d_G^{c^h,c^e} = 686.94 - 414.27 = 272.66$.
\end{itemize}

We have chosen this example because $d_G^c + d_G^{c^h} = 43.84 + 201.95 = 245.79 < d_G^{c^h,c^e} = 272.66$, that is, costs $d_G$ do not always follow the triangle inequality, even if they sometimes do. Without Equation 27, the optimal result of our allocation MILP would be $[x_G^c] = \begin{bmatrix} x_G^c & x_G^{c^h} \\ x_G^{c^h} & x_G^{c^e} & x_G^{c^h,c^e} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ which would mean that $g$ would keep $c^e$ and add $c^h$ into his allocation. The problem is that this allocation would be found by wrongly assuming a cost of $\sum_{a \in A} \sum_{c \in C} \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} d_G = d_G^c + d_G^{c^h} = 245.79$ only, while this allocation would actually cause the larger increase $\sum_{a \in A} \sum_{c \in C} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} d_G = d_G^{c^h,c^e} = 272.66$. In other words, visiting two individual cities may have a different cost (i.e., smaller, equal to or larger than) than visiting the bundle of these two cities. As a result, Equation 27 forbids this allocation with the contraint $x_G^c + x_G^{c^h} + x_G^{c^e} \leq 1$.

\textbf{Difference to our previous article:} The allocation of cities in Message 4 in P2P$_b$ is simpler. It considers only what we call Case 2 ($g$ keeps $c^h$ and $h$ receives $c^e$) and Case 3 (no exchange), and ignores bundles.

2.6 \textbf{Mechanism CnP$_b$}

CnP$_b$ is a decentralised auction in which one of the agents plays the role of host/auctioneer, and the others – when they agree to take part in the considered round – are guests. Figure 3 shows the three messages exchanged between a host $h$ and $G \leq (m - 1)$ guests ("minus one") removing the host) called $g_1, g_2, \ldots, g_G$. These three messages may be described as follows:

1. One salesman becomes the host $h$ in this round by proposing City $c^h$ to the $(m - 1)$ other agents.

2. Each of the $G \leq (m - 1)$ agents interested by $c^h$ becomes Guest $gi$ by replying City $c^h$ and his costs $d_G^e, d_G^{c^h}$ and $d_G^{c^h,c^e}$. Like in P2P$_b$, $c^e$ is found by solving the modified MTSP in Equations 13–24.

   The salesman not interested in $c^h$ reply $\emptyset$ and will be ignored until the start of the next round. To make Figure 4 easier to read, we assume that these agents are $g(G + 1), \ldots, gm$. They do not appear in the figure.

3. After the reception of all the cities and costs sent by the guests, $h$ computes the optimal allocation of cities and sends it to the guests. For every guest $gi$, this message contains either zero city (e.g., "W" in Message 3 in Figure 4), or one city (e.g., "or $c^h$ or $c^e$" in the figure), or two cities (e.g., "or ($c^h$ and $c^e$)"). For that purpose, $h$ solves the allocation MILP in Equations 25–29 like in P2P$_b$ but with the following sets and matrices:

\begin{itemize}
\item $G \geq 1$ depends on the number of agents not replying $\emptyset$ in Message 2.

\item $A = \{h, g_0, g_1, \ldots, g(G - 1), g_G\}$.

\item $C = \{c^h, c^e, e^e_1, \ldots, e^e_{G-1}, c^0_G\}$.

\item $C = \{c^h, c^e, e^e_1, \ldots, e^e_{G-1}, c^0_G, (c^h, e^e_1), \ldots, (c^h, e^e_{G-1}), (c^e, c^e_1), \ldots, (c^e, e^e_{G-1})\}$.

\item \( [d_G^e] = \begin{bmatrix} d_G^h & d_G^{e_1} & \cdots & d_G^{e^e_{G-1}} & d_G^e \\ d_G^{e^e_0} & d_G^{e^e_1} & \cdots & d_G^{e^e_{G-1}} & d_G^e \\ d_G^{e^e_1} & \cdots & \cdots & \cdots & d_G^e \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ d_G^{e^e_{G-1}} & \cdots & \cdots & d_G^{e^e_{G-1}} & d_G^e \end{bmatrix} \)

\item \( [d_G^{c^h,c^e}] = \begin{bmatrix} d_G^{c^h,c^e_{G-1}} & \cdots & \cdots & \cdots & d_G^{c^h,c^e} \\ d_G^{c^h,c^e_{G-1}} & \cdots & \cdots & \cdots & d_G^{c^h,c^e} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ d_G^{c^h,c^e_{G-1}} & \cdots & \cdots & d_G^{c^h,c^e_{G-1}} & d_G^{c^h,c^e} \end{bmatrix} \)
\end{itemize}

Clearly, our choice is for guests to compute not all the possible values of $[d_G^e]$, but only the cost of the two-city bundle containing both $c^h$ and the city proposed by the considered guest. \( \infty \) replaces non-calculated costs as these cities should not be given to this agent. In practice, we replace $\infty$ by twice the largest (non-infinite) value in Matrix $[d_G^e]$. The values in $[d_G^{c^h,c^e}]$ appear in the same order as in $C$.

$d_G^e$ has $G + 1$ columns and $1 + 2 + G$ rows ("1" for the first line representing $c^h$, plus $G$ rows showing

\footnote{We may imagine here two other possible mechanisms for Organisation CnP$_b$. The first would make guests compute all values in $[d_G^e]$ such that $\infty$ would no longer be necessary. The second alternative would consider all two-city bundles in $C$ (e.g., $c^h,e^e_1$) is not considered in our version of the mechanism), which would thus make Matrices $[d_G^e], [c_G^e], [e_G^e]$ and $[x_G^e]$ much larger.}
Costs $d^g_i$ for single cities, and the last $G$ rows for distances $d^g_{i,v^j}$ for two-city bundles $(c^g, c^G)$. 

\[
\begin{array}{cccc}
1 & 1 & 1 & \ldots & 1 & 1 \\
1 & 1 & 0 & \ldots & 0 & 0 \\
1 & 0 & 1 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
1 & 0 & 0 & \ldots & 1 & 0 \\
1 & 0 & 0 & \ldots & 0 & 1 \\
1 & 0 & 1 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
1 & 0 & 0 & \ldots & 1 & 0 \\
1 & 0 & 0 & \ldots & 0 & 1 \\
\end{array}
\]

The values in $[w^g_i]$ appear in the same order as in $C$.  

**Difference with our previous article:** The same messages are exchanged in CNP$_b$, but with less information in Message 2. In fact, $g_i$ sends “$\emptyset$ xor ($c^g_i$ and $d^g_i$)” in CNP$_b$, instead of “$\emptyset$ xor ($c^g_i$ and $d^g_i$) and $d^g_{i,v^j}$.” Bundles are not considered in CNP$_b$: $g_i$ sends $\emptyset$ when he does not want City $c^g_i$, otherwise he sends City $c^g_i$ and Cost $d^g_i$ when the exchange of cities would reduce his personal route length. In the latter case, $h$ accepts the exchange only if this reduces his individual route length as well.

### 2.7. Mechanism Auction$_b$

Auction$_b$ is similar to CNP$_b$, except that the role of the auctioneer is not played by one of the salesmen, but by CA. Figure 4 shows the interactions in several rounds of interaction. To make this figure easier to read, we assume again that agents $g0, \ldots, gG$ agree to participate in this round by not replying $\emptyset$ in Message 2, and the other $(m - G)$ agents reply $\emptyset$ and are hence ignored until the start of the next round.

- $A = \{g0, g1, g2, \ldots, g(G - 1), gG\}$.
- $C = \{c^g_0, c^g_1, c^g_2, \ldots, c^{G-1}_G, c^G\}$.
- $\hat{C} = C \cup \{c^g_i, c^g_j, \forall i < j\}$, hence $\hat{C}$ has $G + G \times (G - 1)/2$ rows.

The first $G$ rows in this vector represent single cities, and the other $G \times (G - 1)/2$ rows represent two-city bundles, hence $\hat{C}$ has $G + G \times (G - 1)/2$ rows. Let us explain the reason for $G + G \times (G - 1)/2$ rows: It is easy to check that there are $1$ two-city bundle when there are $G = 2$ guests, $3$ bundles when $G = 3, 6$ when $G = 4, 10$ when $G = 5$, and so forth, and $G + G \times (G - 1)/2$ two-city bundles for $G$ guests. That is, for $G$ guests, the two-city bundles are the same as for $(G - 1)$ guests, plus $G$ additional two-city bundles containing guest $G$’s City $c^G$. Hence, the number of two-city bundles equals the sum of all integers between $1$ and $G$.

- $[d^g_i] = \begin{bmatrix} d^g_{11} & \ldots & d^g_{1G} \\
\vdots & \ddots & \vdots \\
d^g_{G1} & \ldots & d^g_{GG} \end{bmatrix}$

Like $\hat{C}$, $[d^g_i]$ has $G + G \times (G - 1)/2$ rows. This matrix also has $G$ columns.

In our version of Auction$_b$, each guest $g_k$ sends “[d$^g_i$ and d$^g_{i,v^j}$, $\forall gi \neq gk$]” in Message 4, that is, he proposes a cost for all other agent’s one-city bundles, but not for all their two-city bundles, i.e., $g_k$ only proposes a cost for two-city bundles in which one of both cities is $c^g_i$. Like in CNP$_b$, the missing costs in $[d^g_i]$ are replaced by $\infty$.

---

1 We may imagine other mechanisms for Organisation Auction$_b$. Basically,
be formed with the cities submitted in Message 2.

## 3. Mechanisms

### 3.1. Experiment 1: Study of the impact of the level of (de-)centralisation on efficiency

In Organisation Cluster$_b$, (i) the $m$ salesmen send all their cities to CA, then (ii) CA creates $m$ clusters with these $n$ cities and sends this allocation back to each salesman who, eventually, (iii) locally solves the TSP in Equations 1-6 with the cities allocated to him. We consider two MILP used by CA to create

the guests may compute and send the costs of all two-city bundles in Message 4, such that $d_{ij}$ would contain all costs and no so. Next, we may consider three-city bundles, four-city bundles, …, and even all the possible bundles which may be formed with the cities submitted in Message 2.

Figure 5: Interactions in every round of Auction$_b$.

3. Experimental comparisons

This section shows two experiments. Subsection 3.1 re-produces the same experiment as in our previous article, to establish whether the results are the same as in the “benevolent” version of our mechanisms. We see that this is not the case, and these experiments thus show that the level of (de-)centralisation is not the only determinant of efficiency. Subsection 3.2 presents a second experiment investigating when (de-)centralisation is most efficient. For this purpose, we gradually reduce the period of computation time.

We call “instance” the location of the $n$ cities and their initial allocation to the $m$ salesmen. The instances are different in both experiments, to check that they have no impact on the results. Both sets use the Euclidean distance.

### 3.1. Experiment 1: Study of the impact of the level of (de-)centralisation on efficiency

The first experiment addresses the following question: if a mechanism $m_1$ for MTSP$_b$ is better than another $m_2$, then is its “benevolent” version $m_1_b$ also better than $m_2_b$? In other words, is the level of (de-)centralisation the only feature imaging performance? We explore this question by comparing the results of the mechanisms detailed in this article with those in our previous paper.$^4$ We see that the answer is negative, because the ranking of mechanisms is different in MTSP$_b$ and MTSP$_s$.

$^4$As noted above, this article is complete within itself; accordingly, our older results are described.
3.1.1. Settings

In the first experiment, the instances with \( m = 9 \) salesmen are the same as in our previous article: Each city is located at the same place and allocated to the same salesman (same initial endowment), as detailed below. Conversely to our previous work which also showed the results for \( m = 5 \) salesmen, we think it more informative to show the results for \( m = 2 \) instead, so that the difference between these two sizes of instances is bigger. We compare our mechanisms on the same samples of 130 instances generated by the circular permutation of the ordinate of the 130 cities in “CH130” from TSPLIB ([http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/ch130.tsp.gz](http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/ch130.tsp.gz)) and use only the first \( n \) of these cities. That is, the \( i \)th city in our \( i \)th instance has the same abscissa as the \( i \)th city in CH130, and the same ordinate as the \( ((c+i)\%130) \)th city in CH130 (where “\%130” represents the rest of the division by 130). The initial endowment is also obtained by circular permutation: \( (\text{City 0 is the depot allocated to all salesmen, and}) \text{City 1 is allocated to Salesman 0, City 2 to Salesman 1, ...} \text{City} m \) to Salesman \( m - 1 \), City \( m + 1 \) to Salesman 0, City \( m + 2 \) to Salesman 1, and so on until City \( n - 1 \). As a consequence, the \( i \)th city in the \( i \)th instance with \( n \) cities has the same location and is initially allocated to the same salesman as the \( i \)th city in the \( i \)th instance with \( n + 1 \) cities and, of course, the latter instance has one additional city.

Each point in Figure 6 is the ninth decile (top) or median (bottom) over these 130 instances of the ratios of the route length found by the considered mechanism in less than 30 minutes of computation divided by the length found by Mechanism Centr\(_b\) in the same time span. Shortly, the 130 instances are solved seven times (once per mechanism), then the total route length found by one mechanism is divided by the length found by Centr\(_b\) and, finally, a decile of these 130 ratios is plotted as a point in Figure 6.

The time limit of 30 minutes takes into account concurrency. For instance, each of the two salesmen in P2P\(_b\) may compute for up to 30 minutes. As another example, in CNP\(_b\), if CA needs 15 minutes to create clusters, then each salesman has up to 15 minutes to locally solve the TSP in Equations 15-24. Section 4 and Subsection 5.3 in our previous article detail this point and everything is exactly the same here.

The results in this subsection were obtained on PCs with Intel Core i7-10700 CPU @ 2.90 GHz and 16 GB RAM on 64-bit Windows 10 running AnyLogic 8.7.2 (configured to use 4,096 Mb of memory) and CPLEX 12.10.

3.1.2. Results

The results in Figure 6 show that the best mechanism is always Centr\(_b\) because all the (ninth deciles and medians of the) ratios comparing every mechanism to Centr\(_b\) are larger than 1 for all the considered sizes of instances. To be precise, for larger instances and \( m = 2 \) salesmen, the median (respectively, the ninth decile) of the total route lengths of Auction\(_b\), CNP\(_b\), and P2P\(_b\) reaches a plateau \( \approx 10\% \) (respectively, \( \approx 20\% \)) longer than Centr\(_b\). In the same way, the medians of Cluster\(_R_b\) and Cluster\(_S_b\) are \( \approx 17\% \) worse (longer) than Centr\(_b\) for \( m = 2 \), and their ninth deciles are \( \approx 30\% \) worse. The medians of NoRealloc\(_b\) are \( \approx 40\% \) worse and its ninth deciles \( \approx 50\% \) worse. However, such a plateau has not been reached yet for \( m = 9 \), but we can see for 6 cities per salesman (i.e., \( n = 55 \) cities) that the medians of Auction\(_b\), P2P\(_b\), CNP\(_b\), Cluster\(_R_b\)/Cluster\(_S_b\) and NoRealloc\(_b\) are 25%, 35%, 45%, 45% and 60% worse respectively than Centr\(_b\). The corresponding ninth deciles are 38%, 50%, 65%, 68% and 78% worse.

We may conclude from these numbers that, conversely to our previous article, our results no longer show that more centralisation induces more efficiency because, for example, P2P\(_b\) is at the same time more decentralised and more efficient than CNP\(_b\), Cluster\(_R_b\) and Cluster\(_S_b\).

Another way to state this result is to note that the rankings (i.e., relative efficiencies) of our mechanisms are not the same with their “benevolent” and “selfish” versions. Table 1 summarises the rankings in our previous paper, in this experiment and that in the next subsection. What we have just observed in Figure 6 is summarised on the right-hand side of the first row in this table: Centr\(_b\) performs much better than Auction\(_b\), which performs slightly better than P2P\(_b\), ..., Cluster\(_R_b\) and Cluster\(_S_b\) are equivalent, and NoRealloc\(_b\) has the worst performance. The next subsection will obtain the same ranking for other instances when the time limit is sufficient for Centr\(_b\) to find the optimum (over 28 seconds).

| Variant of MTSP | Time limit |
|------------------|------------|
|                  | too short for Centr\(_b\) to obtain optimum | long enough for Centr\(_b\) to obtain optimum |
| Cluster\(_R_b\)  | Centr\(_b\) | Cluster\(_R_b\) |
| > Auction\(_b\) | => Auction\(_b\) | > P2P\(_b\) |
| ≈ P2P\(_b\)     | ≈ CNP\(_b\)  | ≈ Cluster\(_R_b\) |
| > Cluster\(_S_b\)| ≈ Cluster\(_S_b\) | ≈ Cluster\(_R_b\) |
| => NoRealloc    | => NoRealloc | => NoRealloc |
| Centr\(_b\)     | => Cluster\(_R_b\) | => Cluster\(_R_b\) |
|                  | [all \( n \) in Fig. 6] | [all \( n \) in Fig. 6] |
|                  | [\( n = 121 \) and time limit \( < 7 \) s. in median in Fig. 7] | [\( n = 121 \) and time limit \( > 28 \) s. in Fig. 9 in Moyaux and Marcon (2020)] |

Table 1: Ranking of the mechanisms for \( m = 9 \) salesmen (\( => \) means “is a much better mechanism than”, \( > \) “is better than” and \( \approx \) “is as good as”)
Figure 6: Experiment 1: Deciles of the ratios of quality of every mechanism compared to Mechanism Centr for $m = 2$ (left) and $m = 9$ (right) salesmen and $m$ cities generated from TSPLIB-CH130 with a time limit of 30 minutes.
By contrast, the right-hand side of the second row in Table 1 summarises the fact that we had another ranking with the “selfish” version of our mechanisms. This ranking shows that more centralisation induces better efficiency for MTSP\(^b\). In other words, our older results for MTSP\(_s\) support the widespread intuition that centralisation is best when there is no issue of reactivity.

If we compare the two columns in the second row in Table 1, we may conclude that adding a constraint modelling the selfishness of agents may make Centr\(_b\) the best mechanism for small instances (less than 5 cities per salesman), and the worst for larger instances when the time frame is too short to find the optimal solution.

We have just had a look at three of the four entries in Table 1. The left-hand side of its first row shows the rankings when Centr\(_b\) does not have enough time to find the optimal solution. The next subsection shows where this information comes from.

### 3.2. Experiment 2: Search for the time limit favouring decentralisation

The experiment presented in this subsection addresses the following question: when does decentralisation outperform centralisation with our mechanisms for MTSP\(_s\)? Or, more technically, below what time limit do P2P\(_b\), CNP\(_b\), Auction\(_b\) and/or Cluster\(_b\) outperform Centr\(_b\)?

For instances small enough for Centr\(_b\) to complete its optimisation and hence find the best (shortest) route, decentralised mechanisms cannot find a better solution as it is the optimal one. In our previous paper, we saw that all the tested decentralised mechanisms found better solutions than Centr\(_b\) when the time limit prevents Centr\(_b\) from completing its optimisation, as just seen on the left-hand side of the second row in Table 1. Hence, the following experiments gradually modify one of the three interconnected levers which may make (de-)centralisation the best option:

- **m and n**: These two parameters change the size of the instances. The larger \(n\), the larger the computation time of all mechanisms. For a given \(n\), increasing \(m\) favours decentralisation because the salesmen locally solve their TSP with fewer cities.

- **Time limit**: Since our first experiment sets the maximum duration of computation at 30 minutes and makes \(m\) and \(n\) vary, this second experiment does the opposite by gradually reducing the time limit.

#### 3.2.1. Settings

Experiment 1 used TSPLIB-CH130 to generate instances of up to 130 cities. We first thought it would be useful in Experiment 2 to have as many cities as we wished (which turned out to be useless) and, also, to investigate whether the experimental results change when another method is used to generate instances. For both these reasons, we have now changed the considered instances by assuming that the places to be visited are clients in a city – even if we keep calling these places “cities” and using the Euclidean distance. For this purpose, we first set the seed of the Pseudo Random Number Generator of AnyLogic ("getEngine().getDefaultRandomGenerator().setSeed(main.instance)"); where Parameter “main.instance” is an integer in the “main” of AnyLogic; The value of this parameter is the same for all mechanisms when they solve this particular instance. Next, we generate the location of each city in polar coordinates ("double r=normal(150,0)"); and "double theta =uniform(0,360)");, then translate them into AnyLogic’s Cartesian coordinates ("setXY(main.rectangleWidth/2+r*cos(theta), main.rectangleHeight/2+r*sin(theta))");. We generate 26 instances (instead of 130 in Experiment 2) of \(n = 121\) cities (instead of the quantity indicated on the horizontal axis in Figure 6) i.e., between \(n = 10\) and \(n = 55\) for \(m = 9\) salesmen). Thus, every point in Figure 7 is the (fifth or ninth) decile of the total route length found by a given mechanism running 26 times (once per value of “main.instance”) divided by the total route length found by Centr\(_b\) on these same 26 instances. This figure shows the two successive parts in Experiment 2 together:

1. **Screening**: We start with a time limit of 30 minutes (1,800 seconds) and gradually divide it by two times ten times. Hence, the horizontal axis in Figure 7 is logarithmic. (Technically, in a “Parameter Variation” experiment, AnyLogic provides the integer “index” to identify what iteration it is simulating. We set Parameter “main.instance” in our model to “(int)(index/10)” and Parameter “main.maxComputationTimeSpan” to “1800000/Math.pow(2,index%10)”).

2. **Detailed**: The points obtained in the screening between 3.5 seconds and 14 seconds are close to 1. This means that Centr\(_b\) is the worst mechanism when the time limit is below 3.5 seconds, and the best when this limit is greater than 14 seconds. Consequently, we conduct more exhaustive experiments between these two limits, starting at 14 seconds and progressively decreasing by 1.25 seconds ten times. (Technically, Parameter “main.instance” is the same as in the previous paragraph, but “maximumComputationTimeSpan” is set to “1800000/Math.pow(2,7) - 1250*(1+index%10)”.)

The softwares and hardwares in Experiment 2 are the same as in Experiment 1, except that we updated CPLEX to version 20.10.

#### 3.2.2. Results

The first thing to notice in Figure 7 is that the ranking of the mechanisms for time limits between 28 seconds and 30 minutes is the same as in Experiment 1. Consequently, changing the location of the cities has not affected the ranking. Therefore, the right-hand side of the first row in Table 1 is also true for Experiment 2.

The points for time limits below 7 seconds provide a ranking for the left-hand side entry of the first line in Table 1 for

\[5\] Since the ninth deciles of ClusterR\(_b\) and ClusterS\(_b\) are sometimes larger than the deciles of NoRealloc, we write 28 seconds instead of the above 14 seconds.
Figure 7: Experiment 2: Deciles of the ratios of quality of every mechanism compared to Mechanism Centr_b for \( m = 9 \) salesmen and \( n = 121 \) cities generated from a normal distribution.

which our previous article and Experiment 1 have no data. We only look at medians here because, for the first time in our experiments on both MTSP_b and MTSP_s, the ninth deciles give a different ranking than medians. It is also the first time that ClusterR_b and ClusterS_b incur different efficiencies. More precisely, ClusterR_b is the best of all mechanisms, and ClusterS_b is the worst decentralised mechanisms even if it is still better than CentrS_b.

We find once again that our decentralised mechanisms are either all better or all worse than the centralised mechanism Centr_b, except in the tiny zone between 7 and 28 seconds where this is not so clear cut.

Understanding the difference of rankings between the selfish and benevolent versions of our mechanisms is important as it contradicts the intuition that centralisation causes (or, at least, is correlated with) efficiency. We assume that this gap is due to the fact that two-city bundles are not taken into account by any of our selfish mechanisms. From a more general point of view, despite a same level of (de-)centralisation and even an identical AnyLogic state chart, benevolent and selfish versions of a same mechanism may operate very differently, as pointed out in Footnotes 2 and 3 which suggest variants of two of our mechanisms.

4. Conclusion

This article contributes to understanding when and how to (de-)centralise decision making in organisations. For this purpose, it compares mechanisms with various levels of (de-)centralisation and that solves a Multiple Travelling Salesmen Problem (MTSP) in order to explore when and how to (de-)centralise allocation. We study the efficiency of decisions when time is limited because the advantage of centralisation is said to be its capability to guarantee optimality in the absence of such a time limit, while the advantage of decentralisation would be its reactivity when this limit is tight. We compare the efficiency of our mechanisms with one another, and also refer to the same comparison on the variant of MTSP in a previous article of ours in which the salesmen were assumed to be selfish. Our results indicate that the relative efficiency of the mechanisms is different in the traditional MTSP and the variant with selfish salesmen. Our results with the traditional MTSP are therefore not as intuitive as what we obtained with this variant, in which a higher level of centralisation caused (or was just correlated with) a greater efficiency. For both versions of MTSP, we think that an important conclusion is that our fully centralised mechanism – which we call Centr – is either better than all our decentralised mechanisms, whatever their level of decentralisation, or worse than them all, and the zone in between is very narrow. In other words, Centr is the best when the time frame is long enough to find the optimum or to approximate it, but suddenly becomes the worst when we gradually reduce the time available to make decisions. Our results also highlight the fact that various mechanisms may be instances of a same decentralised organisation, and various mechanism designs for a same organisation may yield very different efficiencies.
This article addresses two versions of MTSP. In future work we would like to explore the impact of the level of (de-)centralisation in a third variant of MTSP. We are interested in models for food delivery, to check if centralisation is more efficient in terms of kilometres travelled by delivery bikers, than a decentralised mobile application. If centralisation is best, then the difference of efficiency would provide insights into the value provided to bikers by companies currently operating such services.

5. acknowledgements

The author thanks the department of Industrial Engineering of INSA-Lyon for the use of the computers in one of its student laboratories to generate the large quantity of experimental data summarised in this article.

6. Conflict of interest

The author declares that he has no conflict of interest.

References

[1] Bektas T (2006) The multiple traveling salesmen problem: An overview of formulations and solution procedures. Omega 34:209–219
[2] Cardin O, Trentesaux D, Thomas A, Castagna P, Berger T, El-Haouzi HB (2017) Coupling predictive scheduling and reactive control in manufacturing hybrid control architectures: state of the art and future challenges. Journal of Intelligent Manufacturing 28(7):1503–1517, DOI 10.1007/s10845-015-1139-0
[3] Davidsson P, Persson JA, Holmgren J (2007) On the integration of agent-based and mathematical optimization techniques. In: Carbonell JG, Siekmann J (eds) Proc. 1st KES Int. Symposium, Agent and Multi-Agent Systems: Technologies and Applications, Lecture Notes in Computer Science 4406 (Springer), Wroclaw, Poland, pp 1–10
[4] Derigent W, Cardin O, Trentesaux D (2020) Industry 4.0: Contributions of holonic manufacturing control architectures and future challenges. Journal of Intelligent Manufacturing DOI 10.1007/s10845-020-01532-x
[5] Kivelevitch E, Cohen K, Kumar M (2013) A market-based solution to the multiple traveling salesmen problem. J Intel & Robotic Sys 72:21–40
[6] Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesmen problems. J Association of Computing Machinery 7:326–9
[7] Moyaux T, Marcon E (2020) Cost of selfishness in the allocation of cities in the multiple travelling salesmen problem. Engineering Applications of Artificial Intelligence 89, DOI 10.1016/j.engappai.2019.103429
[8] Moyaux T, McBurney P (2012) Centralised vs. market-based and decentralised decision-making: A review. Cybernetics & Systems 43(7):567–622
[9] Rao MR (1971) Cluster analysis and mathematical programming. J of the American Statistical Association 66(335):622–626
[10] Saliez Y, Berger T, Raileanu S, Chaabane S, Trentesaux D (2010) Semi-hierarchical control of FMS: From theory to application. Eng Appl Intell Intell 23:1314–1326
[11] Sağlam B, Salman FS, Sayın S, Türkay M (2006) A mixed-integer programming approach to the clustering problem with an application in customer segmentation. Eur J of Operational Research 173:866–879
[12] Smith RG (1980) The contract net protocol: High level communication and control in distributed problem solver. IEEE Trans on Computers C-29(12):1104–1113
[13] Toth P, Vigo D (2002) The Vehicle Routing Problem. Society for Industrial and Applied Mathematics, Philadelphia
[14] Zhang J, Ding G, Zou Y, Qin S, Fu J (2019) Review of job shop scheduling research and its new perspectives under industry 4.0. Journal of Intelligent Manufacturing 30(4):1809–1830, DOI 10.1007/s10845-017-1350-2