Align-BDD application: on joint UFLP.

Alexey Bochkarev

June 4, 2022

1 TBD: Summary and questions

This section is an internal discussion, while all the rest I hope to include into the paper draft. Here I tried to get into reviewers' shoes and try to think critical about what are we trying to present. So, here are some weak points I would propose to discuss:

1. **What worries me most:** I figured CPP MIP is still faster. That is, if I build BDDs, but skip aligning the diagrams altogether and just solve the CPP MIP using Gurobi, it just runs way faster. For a specific instance (more or less representative, as I see it) a naive MIP takes 28 sec, CPP MIP works below a second, and VS-heuristic needs about 3 sec (to-A baseline takes 5 or 14 sec., depending on what we choose for 'A'). Overall, a naive MIP tends to be the slowest, alternative alignment heuristics are somewhat faster than MIP, but slower than our VS-heuristic, and CPP MIP is clearly the fastest. Maybe that is because the diagrams are small, but this is what I see from my experiments. It may or may not get better if we try to design instances where I have still limited, but larger number of connection points between any two clusters (e.g., no more than 3 arcs between any two clusters). I would need to modify the algorithm, and I am not sure if that would help, to be honest. From the other hand, this example just shows: if we are OK with MIP in the end, we enjoy faster runtimes. With our heuristic, we could compromise some runtime, but have an LP in the end. I am still not sure if that would count as a proper response to the reviewers :) although it is indeed best we could get by this point. A separate question is how to present this carefully, if we choose to proceed with this problem.

2. Another somewhat weak point (although perhaps it is no different from the previous one) is that we are feeding more information to the DD-based methods. Viz., I supply pre-defined data that explicitly says what points are in which cluster, and what are cluster's connection points. I understand that this is sort of the whole point, that we are designing a heuristic that would use additional information that the generic solver doesn't know how to use. But I wanted to discuss: do you think this comparison is fair? Or rather, how to describe it in a correct way, so that we wouldn't mislead the reader.

3. I was looking at Figure 1 and trying to see if I could just reformulate this problem as a larger UFLP instance (or something along those lines). It seems I can not, so it is indeed a separate, more complex problem, right?

4. Finally, I have not that much variety in generated graph topologies: it is always n clusters of M points, the number of edges per cluster is always the same. I think for $M = 5$ nodes per cluster, it is not that much. I am not quite sure if this is a problem.

2 Joint UFLP formulation

We next illustrate the performance of our algorithms in the context of a specific application. In this section we introduce the problem, then discuss a naive MIP formulation and an alternative CPP approach (where our heuristic can be used to align the diagrams) in Section 3, and conclude with numerical experiments in Section 4.

We have designed a problem to highlight a possible application for our heuristic: It comprises two similar subproblems of special structure, which makes it natural to represent with BDDs, linked with side constraints implying the interdependence of some decisions across two subproblems. A subproblem is the following modification of the uncapacitated facility location problem, UFLP (Owen and Mark S. Daskin, 1998; ReVelle, Eiselt, and M. S. Daskin, 2008).

Each of the two subproblems (indexed with $t = 1$ or 2) considers a set of N points. At each point i, we can locate a facility at a cost given by c_i^t, covering all points in a set given by S_i^t, where $i \in S_i^t$. Set S_i^t might refer to customers that are sufficiently close to location i according to some specified metric, like distance or travel time. Therefore, the N points represent some graph G_t and

S_i^t encode the respective adjacency lists. (For convenience, we assume G^t to be connected.) We also define overlap cost function $f_i^t : \mathbb{N}_0 \to \mathbb{R}$, so that $f_i^t(a)$ would indicate a cost of covering point i exactly a_i times, within subproblem t. Note that we do not imply any properties of f_i, such as convexity or concavity. Connection between the subproblems is given by the condition that there is a set of pairs of points $(i_1, i_2) \in J$ such that the location decision regarding point i_1 in G_1 must coincide with the corresponding decision on i_2 in G_2. The problem, which we refer to as *joint UFLP (j-UFLP)*, minimizes the total cost and can be formulated as follows:

$$\min \sum_{i=1,t=1,2}^{N} \left(c_i^t x_i^t + f_i^t(a_i^t) \right) \tag{1a}$$

$$\text{s.t. } a_i^t = \sum_{j \in S_i^t} x_i^t \qquad \text{for all } i = 1, \ldots, N, t = 1, 2, \tag{1b}$$

$$x_i^t \in \{0, 1\} \qquad \text{for all } i = 1, \ldots, N, t = 1, 2, \tag{1c}$$

$$x_i^1 = x_j^2 \qquad \text{for all } (i, j) \in J. \tag{1d}$$

Further, to make the problem natural to represent with BDDs, we assume that each sub-problem graph consists of n subgraphs (*clusters*), M nodes each, and that there is at most one arc between every two subgraphs (we refer to the endpoints of such arcs as *connection points*). Moreover, we assume J to link the connection points only. An example of such instance is presented in Figure 1. A subgraph corresponding to the first subproblem, G_1, is depicted with thick lines, while G_2 is drawn with thin lines. Linking conditions (1d) imply that some nodes are common in G_1 and G_2. Such nodes are marked with background color (nodes `j1`, `j6`, `j10`–`j12`, `j16`, `j19`, and `j21`). Note that overlaps are calculated independently: e.g., locating a facility at `j10` does contribute to a_{16}^2, but not a_{16}^1.

TBD: ... and after that last sentence I am starting to doubt if the whole idea of representing the two subproblems on a single graph was good. Well, there is an arc between `j10` and `j16`, but it only "works" when I calculate overlaps in one subproblem, not in the other one... I feel the need to depict everything in one picture, but does it makes the exposition easier?

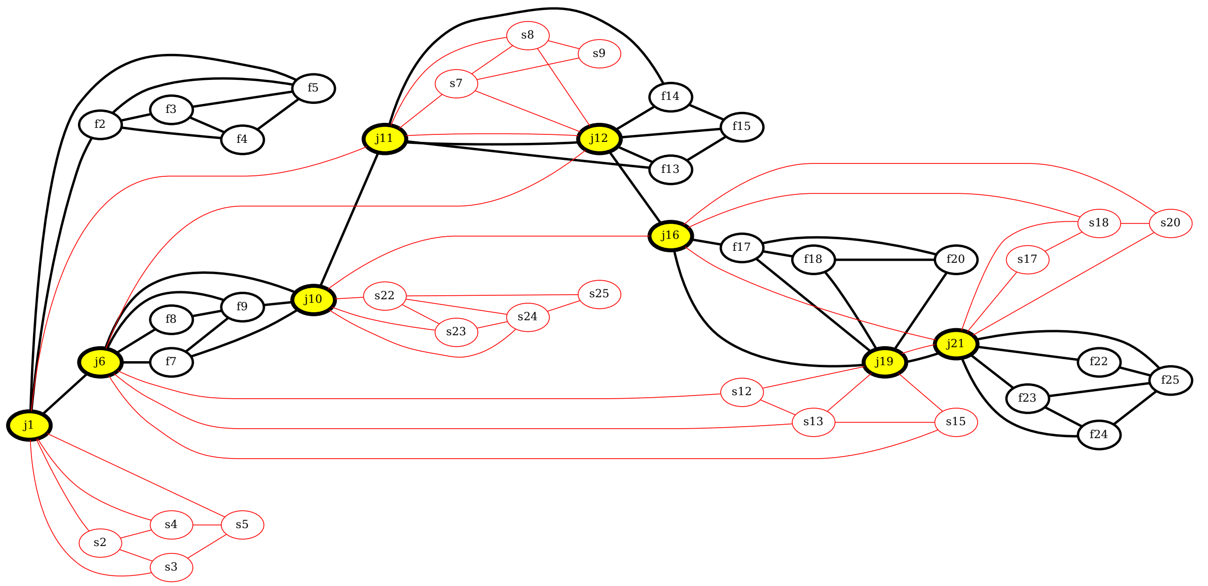

Figure 1: Sample j-UFLP instance graph.

3 Solution methods

A naive MIP reformulation for (1) implies introducing new binary variables $y^t_{i,a}$ indicating whether point i in subproblem t was covered at least a times:

$$\min \sum_{i=1,t=1,2}^{N} \left(c^t_i x^t_i + \sum_{a=1}^{|S^t_i|} q^t_{i,a} y^t_{i,a} \right) + C \tag{2a}$$

$$\text{s.t.} \sum_{a=1}^{|S_i|} y^t_{i,a} = \sum_{j \in S^t_i} x^t_i \qquad \text{for all } i = 1, \ldots, N, t = 1, 2, \tag{2b}$$

$$y^t_{i,a} \geq y^t_{i,a+1} \qquad \text{for all } i = 1, \ldots, N, t = 1, 2, a = 0, \ldots, |S_i| - 1, \tag{2c}$$

$$x^t_i \in \{0, 1\} \qquad \text{for all } i = 1, \ldots, N, t = 1, 2, \tag{2d}$$

$$x^1_i = x^2_j \qquad \text{for all } (i, j) \in J, \tag{2e}$$

where $q_{i,a} = f^t_i(a) - f^t_i(a-1)$ and $C = \sum_{i,t} f^t_i(0)$ are constants.

From the other hand, such problem can be represented as a CPP with two fixed-width diagrams (each corresponding to a subproblem), having different order of variables. For each subproblem, we build a BDD, where a path captures total cost stemming from the corresponding location decisions in the following way. We process the clusters in the order we would traverse them without entering any cluster twice, and add the connection points to the diagram. Note that when all (at most, four) connection points adjacent to a cluster are added to the BDD, we can calculate the cost stemming from that cluster by solving at most four smaller mixed-integer problems. Moreover, in the course of BDD construction due to the special structure of G we need to keep track of the values for at most three variables, which limits the BDD width to $2^3 = 8$. This BDD construction process is illustrated below. Interconnection between subproblems is achieved by renaming the variables in the diagrams: For each $(i, j) \in J$ labels of the BDD layers corresponding to x_i^1 and x_j^2 (in the first and the second BDD, respectively) must coincide.

Example. Let us briefly illustrate the BDD construction procedure for a single subproblem with Figure 2. Assume we have four clusters of points, denoted `C1`, ..., `C4` and points `1`, ..., `6` are connecting points for these clusters. First, observe that when we fix values for x_1 and x_2, the costs contribution to the objective stemming from cluster `C1` can be obtained by solving a smaller mixed-integer program. The formulation is similar to (2), but with index t and linking condition (1d) dropped and the set of nodes restricted to `C1`. Such MIP would have only M x_i-variables, corresponding to `C1`. Therefore, we introduce x_1 and x_2 to the BDD and capture the costs stemming from cluster `C1` in BDD arc weights. The last new layer would then comprise $2^2 = 4$ nodes, for (x_1, x_2) being equal $(0, 0)$, $(1, 0)$, $(0, 1)$, and $(1, 1)$, respectively. If we further add x_3 and x_4 to the diagram, we can fix costs stemming from cluster `C2` in BDD arc costs as well. After we add x_3 the diagram width increases to $2^3 = 8$ (we are keeping track of x_1, x_2, and x_3), but after we introduce x_4 we do not need to have $2^4 = 16$ nodes in the last layer. Note that x_1 and x_2 will not affect any costs stemming from `C3` and `C4`. Therefore, BDD nodes corresponding to different values of these two variables in the last layer can be joined, which would make it sufficient to have x_4 layer with 4 nodes only. Continuing this process, we build a diagram of width $2^2 = 8$ at most (regardless of n and M), that encodes the subproblem. The resulting BDD is presented in Figure 3. Because we drop the information regarding specific choices of x_1 and x_2 the highlighted nodes in Layer 4 will yield only two child nodes in Layer 5. We indicate selected arc costs in the figure, denoting the objective for the cluster subproblem `C1` as $C1(x_1, x_2)$, for `C2` as $C2(x_1, x_2, x_3, x_4)$, for `C3` as $C3(x_3, x_4, x_5, x_6)$, and for `C4` as $C4(x_5, x_6)$. Note that a natural order of variables that allows such compact BDD representation is determined by the sequence of clusters in the figure (we processed them from the first to the fourth). Finally, if, for example, we have another subproblem encoded by a BDD with variables $x'_1, \ldots, x'_6$, and $J = \{(1,6),(2,5),(3,4),(4,3),(5,2),(6,1)\}$, we could reformulate the j-UFLP as a Consistent Path problem instance by renaming the variables in the diagram depicted in Figure 3 from $(x_1, x_2, x_3, x_4, x_5, x_6)$ to $(x'_6, x'_5, x'_4, x'_3, x'_2, x'_1)$.

4 Numerical experiments

We generated approximately 400 instances for each value of n (number of clusters), ranging from 5 to 14. Each cluster contained $M = 5$ nodes, with sparsity parameter $L = 0.25$. The latter implies that during the graph generation we kept adding edges until the number of randomly generated edges $|E_k|$ in the cluster satisfied:

$$L > 1 - \frac{|E_k|}{M(M-1)/2},$$

assuming about one quarter of all possible $M(M-1)/2$ edges were not present. We then solved each instance with each of the two methods:

- First, we used a naive MIP formulation (2) (denoted `MIP` in the figures below).
- Second, we built two BDDs as discussed above. The resulting CPP instance was solved by aligning the diagrams and finding a shortest path between root and terminal nodes. The diagrams were aligned using the proposed variable-sequence based heuristic (denoted `VS-heuristic`) and a simple baseline of aligning the second diagram to match the order of the first one (denoted `to A`).

The results are presented in Figure 4. On the left panel we present runtimes for each of the solved instances, one point per instance. Solution time in seconds (logarithmic scale) is along the vertical axis, number of clusters n is along the horizontal axis. We see that since we are leveraging the additional information regarding the composition of the clusters in the BDD-based approach, VS-based heuristic tends to perform relatively better than the naive MIP, and this gap increases as the instance size grows. (Numbers in the rectangles above the lines in the figure indicate the share of instances where the proposed heuristic were faster than `MIP` at least by 10%.) On the right panel we present histograms of the runtimes for `MIP` and `to A` heuristics, relative to the proposed

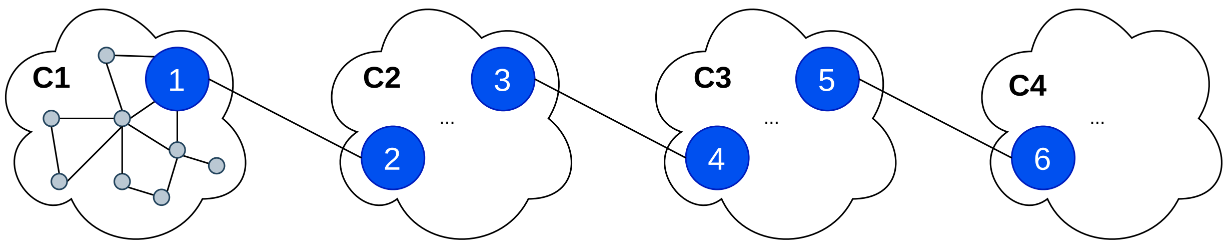

Figure 2: Sample j-UFLP subproblem graph.

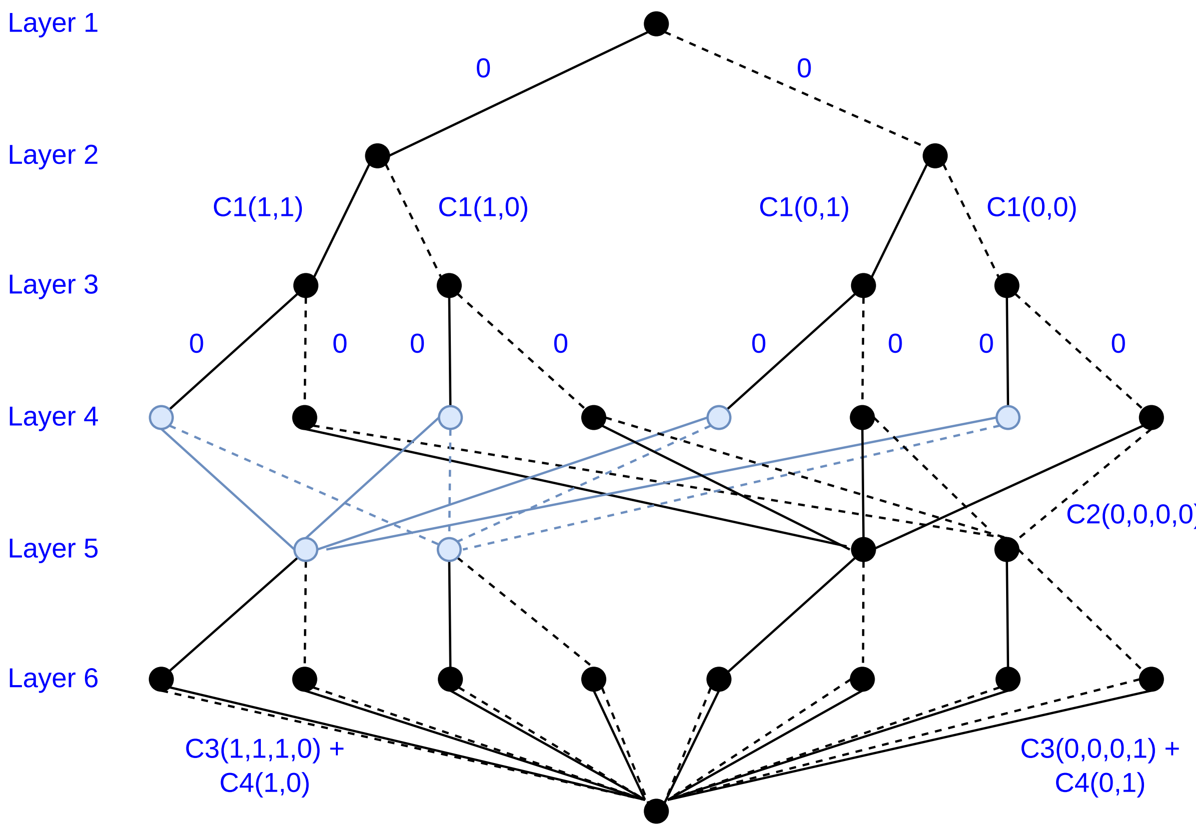

Figure 3: Resulting BDD for the subproblem in Figure 2.

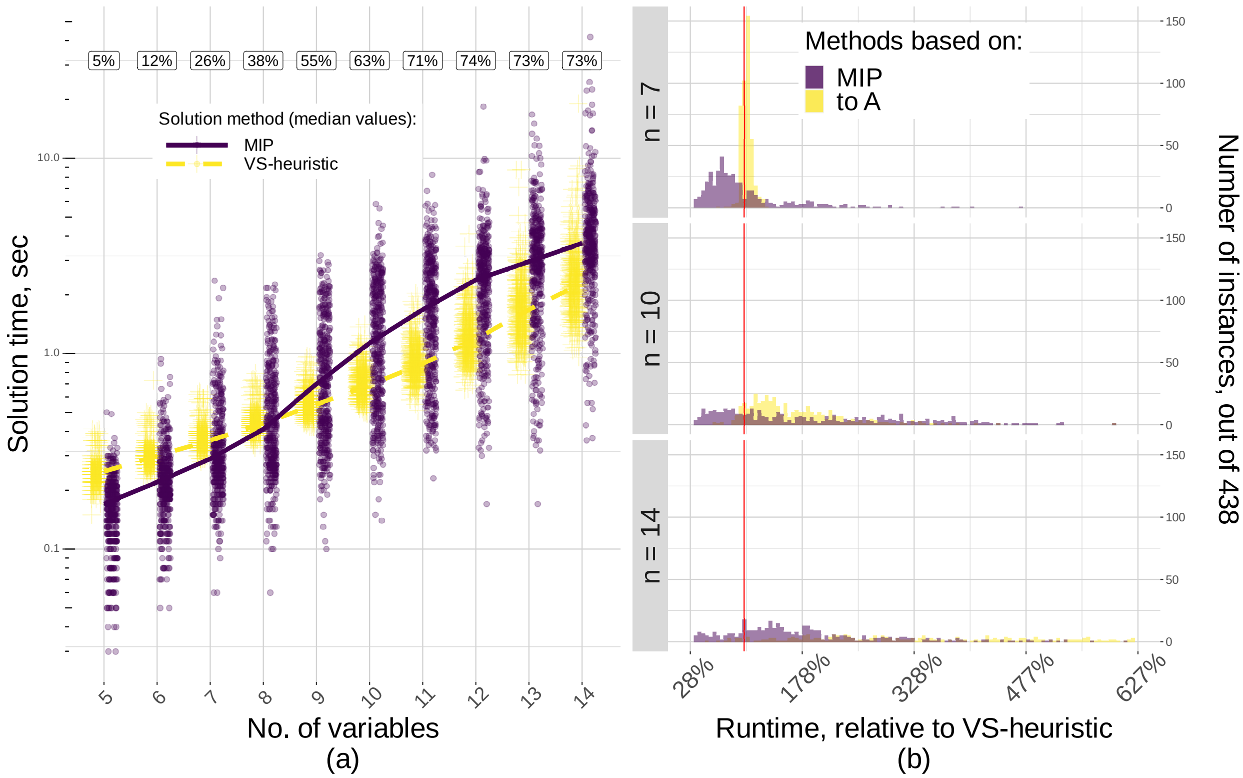

Figure 4: Numerical performance of various heuristics for j-UFLP.

`VS-heuristic` for the different numbers of clusters n. The value of 1.0 (vertical line) implies that the heuristic takes the same time (for a particular instance) as `VS-heuristic`. Values to the left of the vertical line imply the heuristic outperforms the proposed one. We see that while for $n = 7$ clusters `VS-heuristic` is almost always slower than MIP, for larger instances it starts to outperform the baseline on more than two thirds of the instances.

This is perhaps for Conclusion section: This example illustrates a possible way to reformulate a complex combinatorial problem as a large LP. When such problem possesses a special structure that makes it natural to represent with a collection of BDDs with different order of variables, we can try to align these BDDs to build an intersection and formulate a shortest-path in the intersection BDD. The heuristic we propose in many cases allows to find a good shared variable order for the diagrams and avoid costly manipulations with decision diagrams directly.

References

Owen, Susan Hesse and Mark S. Daskin (Dec. 1998). “Strategic Facility Location: A Review”. In: *European Journal of Operational Research* 111.3, pp. 423–447. ISSN: 03772217. DOI: 10.1016/S0377-2217(98)00186-6 (cit. on p. 2).

ReVelle, C. S., H. A. Eiselt, and M. S. Daskin (Feb. 2008). “A Bibliography for Some Fundamental Problem Categories in Discrete Location Science”. In: *European Journal of Operational Research* 184.3, pp. 817–848. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2006.12.044 (cit. on p. 2).