Delayed Disaggregation for Benders Decomposition

Yannik Rist[a] and Michael Forbes[a]

[a] University of Queensland, Brisbane, Australia
yannik.rist@uq.net.au, m.forbes@uq.edu.au * Corresponding author

Extended abstract submitted for presentation at the 11th Triennial Symposium on Transportation Analysis conference (TRISTAN XI)
June 19-25, 2022, Mauritius Island

April 1, 2022

Keywords: Benders Decomposition; Vehicle Routing; Time windows

Introduction

Combinatorial Benders Decomposition (CBD) (Codato and Fischetti, 2006) is a variation on Benders Decomposition Benders, 1962 for Mixed Integer Programs (MIP) where the inclusion of constraints in the linear subproblem is allowed to depend on the master problem variables. CBD is often used in vehicle routing problems to handle timing constraints for vehicle routes (Ropke et al., 2007; Alyasiry et al., 2019; Rist and Forbes, 2021), albeit without an explicit Benders formulation. Disaggregation of the subproblem is used to produce multiple Benders cuts from one master problem solution. Our main contribution is the idea of *delayed disaggregation*; disaggregating the subproblem and optimality cuts after solving the subproblem rather than before. We give an example of why this is useful for vehicle routing subproblems in particular.

Benders Decomposition with Delayed Disaggregation

In CBD, a Benders Master Problem is defined,

$$\min \theta + f^\top y \tag{1}$$

$$\theta \text{ subject to Benders Cuts} \tag{2}$$

$$y \in D \tag{3}$$

where $D \subseteq \{0,1\}^n$ and θ is the estimator of the subproblem objective. The subproblem, containing conditional constraints, is defined below.

$$\min c^\top t \tag{4}$$

$$y_{i_k} = 1 \implies {a^k}^\top t \le b_k \qquad 1 \le k \le K \tag{5}$$

$$t \in \mathbb{R}^m_{\ge 0} \tag{6}$$

For a particular $y^* \in Y$, let $A(y^*)$ denote the matrix formed by the implied subproblem constraints at y^*. If the subproblem is optimal one may find a Minimally Responsible Subset (MRS) of constraints by taking the set of constraints with non-zero dual variables, and inferring which presently-active master variables imply these constraints. Let $Y_{\text{MRS}} \subseteq \{y_1, \dots, y_n\}$ be such a set of master problem variables, and let t^* be the optimal subproblem solution at y^*. The Combinatorial Benders Cut is

$$\theta \ge (c^\top t^*) \cdot \Big(\sum_{y_k \in Y_{\text{MRS}}} y_k - |Y_{\text{MRS}}| + 1 \Big)$$

The right-hand side equal to $c^\top t^*$ when all variables in Y_{MRS} are set to 1 in the master problem and non-positive otherwise. If $A(y)$ is block-diagonal with L blocks, then the subproblem disaggregates into L smaller subproblems. By splitting $\theta = \sum_{i=1}^m \theta_i$, one θ_i per subproblem variable, we can disaggregate the optimality cut into L individual cuts. Let Y^l_{MRS} be an MRS for the l-th subproblem, and let $I^l \subseteq \{1, \dots, m\}$ index the variables that appear in the l-th subproblem. The disaggregated cuts read:

$$\sum_{i \in I_l} \theta_i \geq \Big(\sum_{i \in I_l} c_i t_i^*\Big) \cdot \Big(\sum_{y_k \in Y^l_{\mathrm{MRS}}} y_k - |Y^l_{\mathrm{MRS}}| + 1\Big) \tag{7}$$

To summarise the process for deriving disaggregated cuts, an integral y^* is found, the subproblem corresponding to $A(y^*)$ is decomposed, each component is solved and finally cuts are calculated. We propose switching the second and third steps; to solve a single subproblem, and decompose it afterwards (and therefore disaggregating the cuts) with full knowledge of the optimal solution. Specifically, once $A(y^*)$ is solved we can remove from $A(y^*)$ all rows whose dual variables are 0 to construct an equivalent, reduced subproblem with constraint matrix $A'(y^*)$. However, since $A'(y^*)$ contains fewer rows than $A(y^*)$ it will decompose into smaller block-diagonal blocks. Specifically, for each block submatrix of $A(y^*)$, deleting rows will sub-divide it into one or more sub-blocks, or remove the block entirely. We now present a class of linear programs commonly found in vehicle routing subproblems which work well with this method of decomposition.

Application: Scheduling Subproblems in Vehicle Routing

In vehicle routing, Combinatorial Benders Subproblems are often linear programs (LPs) of the form given below, where $E \subseteq \{(i,j) \mid 1 \leq i, j \leq n\}$ and $a_i, b_i, c_i, d_{ij} \geq 0$.

$$\min c^\top t \tag{8}$$

$$t_i + d_{ij} \leq t_j \qquad \forall\ (i,j) \in E \tag{9}$$

$$t_i \geq a_i \qquad \forall\ 1 \leq i \leq n \tag{10}$$

$$t_i \leq b_i \qquad \forall\ 1 \leq i \leq n \tag{11}$$

$$t \geq 0 \tag{12}$$

Examples of such vehicle routing problems include the Travelling Salesman Problem with Time Windows, Pickup-and-Delivery Problem with Time Windows (PDPTW), and the Active Passive Vehicle Routing Problem (APVRP). One well-known vehicle routing problem which does not fit this classification is the Dial-A-Ride Problem, where $d_{ij} < 0$ for some constraints. The LP (8)–(11) can be represented using a directed graph, called a constraint graph, where nodes represent the variables t_i and edges represent the constraints eq. (9), weighted by d_{ij}, Constraints eq. (9) are called *edge* constraints.

If the constraint graph contains a strongly connected component (SCC) then the SCC implies infeasibility or a simplification of the subproblem. Precisely, if any SCC edge (t_i, t_j) has positive weight then a cycle containing this edge exists which implies $t_i > t_i$. The cycle therefore represents an infeasible subsystem. Otherwise, one can deduce that $t_i = t_j$ for every pair of variables in the SCC and replace the SCC with a single meta-variable. Processing every SCC in this manner, one obtains the condensation graph of the original constraint graph, which is always acyclic.

Solution

It is now assumed that the constraint graph is acyclic, as a result of performing the condensation described above. It is straight-forward to solve the LP represented by the directed acyclic graph. First, all t_i^* are initialised to a_i. Then, traversing the graph in topological order, the successors of node t_i are updated with $t_j^* \leftarrow \max(t_j^*, t_i^* + d_{ij})$. This can be implemented as a modification of Kahn's Algorithm (Kahn, 1962) which runs in $\mathcal{O}(m + |E|)$ time. Note the invariant that t_i^* is an underestimate of the smallest value that t_i can have in *any* feasible solution. Therefore, if any $t_i^* > b_i$ then the LP is infeasible, otherwise the solution is optimal since $c \geq 0$.

Dual Variables and Optimality Cuts

With a small modification to the algorithm above, we can extract the dual variable values from the optimal solution (assuming the problem is feasible). Whenever the update along the edge (t_i, t_j) causes t_j^* to increase, we store t_i as the *active predecessor* of t_j. Each node has at most one active predecessor. If t_i is the active predecessor of t_j, then the edge (t_i, t_j) is said to be *active*. Each set of nodes connected by active edges defines a subgraph which is a tree, since each node has at most one active predecessor. Lonely nodes form their own single-node trees (see figure 1 for an example). This idea is similar to the Critical Path Method of Kelley and Walker (1959).

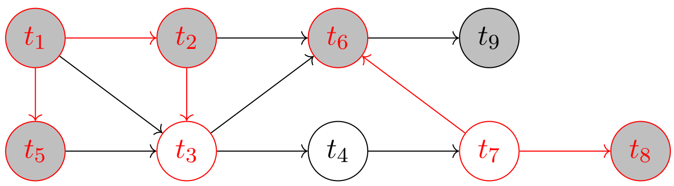

Figure 1 – *Active edges and connected nodes in the constraint graph shown in red. Shaded nodes are t_i with $c_i > 0$*

These trees, whose nodes and edges are denoted by T and E_T respectively, are used to calculate an optimal solution to the dual of (8)–(11). Let $\pi_{ij}, \alpha_i, \beta_i$ denote the dual variables associated with constraints (9)–(11) respectively. For $t_i \in T$, let T_i be subtree of T rooted at t_i. It is straightforward to show that the values,

$$\pi_{ij}^* = \begin{cases} \sum_{k \in T_j} c_k & \text{if edge } (t_i, t_j) \in E_T \text{ is active} \\ 0 & \text{otherwise} \end{cases}$$

$$\alpha_i^* = \begin{cases} \sum_{k \in T} c_k & \text{if } i \text{ is the root of a subtree } T \\ 0 & \text{otherwise} \end{cases}$$

$$\beta_i^* = 0$$

are dual-optimal. LP (8)–(11) has the nice property that block-diagonality in the constraint matrix corresponds to connectedness in the graph. Removing edges with $\pi_{ij}^* = 0$ decomposes the graph into a forest of trees whose edges are active. A Benders Optimality cut of the form (7) can be defined for each tree. Without delayed disaggregation, a cut would be added for each connected component of the original graph.

LP (8)–(11) has another useful property: a solution t^* found by the algorithm described above is optimal for any $c \geq 0$, since it minimises all variables simultaneously. Consider a hypothetical LP for each variable t_i which has the same constraints but a modified objective $c' = c_i e_i$ where e_i is the i-th unit vector. The same t^* is optimal for each of these n subproblems, but the optimal

Table 1 – *Examples of cut disaggregation for figure 1.*

Disaggregation Kind	No. of cuts	Remarks
Regular	1	Graph is connected
Delayed (MRS)	3	The trees are $\{t_1, t_2, t_5\}$, $\{t_7, t_6, t_8\}$, $\{t_9\}$
Delayed (Critical Path)	6	t_i with $c_i > 0$: $\{t_1, t_2, t_5, t_6, t_8, t_9\}$

duals will differ. If we apply delayed disaggregation to each of these problems individually, we find a single *critical path* (Kelley and Walker, 1959) of edges with non-zero duals which ends at t_i. We may therefore can add separated cuts for variables t_i and t_j even if t_i and t_j reside in the same active edge tree in the original LP (the LP with objective c). Note we only need to solve the original LP – the active edges from a variable to the root of its tree form the critical path, which is used to calculate the cut. For example, in figure 1, the active edge tree $T = \{t_6, t_7, t_8\}$ disaggregates into the critical paths (t_7, t_8) and (t_7, t_6). Let p be the critical path for t_i and define Y_p as a set of master problem variables which imply the edge constraints and lower bound at the start of p. Below is the optimality cut which arises from p.

$$\theta_i \geq c_i t_i^* \cdot \Big(\sum_{y_k \in Y_p} y_k - |Y_p| + 1 \Big)$$

Finally, note that if $c_i = 0$ then a cut is not needed. Table 1 summarises the MRS and critical path disaggregations. The former is applicable to any LP subproblem whereas the latter relies on the graph structure of (8)–(11).

Conclusion

We presented a novel enhancement for disaggregation in CBD where disaggregation of cuts is especially important. We showed the utility of the technique for vehicle routing subproblems. Furthermore, we demonstrated how additional assumptions on the subproblem can lead to an even finer level of disaggregation. The conference presentation will contain computational results for the APVRP.

References

Alyasiry, Ali Mehsin, Michael Forbes, and Michael Bulmer (Nov. 2019). “An Exact Algorithm for the Pickup and Delivery Problem with Time Windows and Last-in-First-out Loading”. In: *Transportation Science* 53.6, pp. 1695–1705. DOI: 10.1287/trsc.2019.0905.

Benders, J. F. (Dec. 1962). “Partitioning procedures for solving mixed-variables programming problems”. In: *Numerische Mathematik* 4.1, pp. 238–252. DOI: 10.1007/BF01386316.

Codato, Gianni and Matteo Fischetti (Aug. 2006). “Combinatorial Benders’ Cuts for Mixed-Integer Linear Programming”. In: *Operations Research* 54.4, pp. 756–766. DOI: 10.1287/opre.1060.0286.

Kahn, A. B. (Nov. 1962). “Topological sorting of large networks”. In: *Communications of the ACM* 5.11, pp. 558–562. DOI: 10.1145/368996.369025.

Kelley, James E. and Morgan R. Walker (1959). “Critical-path planning and scheduling”. In: DOI: 10.1145/1460299.1460318.

Rist, Yannik and Michael A. Forbes (Aug. 2021). “A New Formulation for the Dial-a-Ride Problem”. In: *Transportation Science*. DOI: 10.1287/trsc.2021.1044.

Ropke, Stefan, Jean-François Cordeau, and Gilbert Laporte (2007). “Models and branch-and-cut algorithms for pickup and delivery problems with time windows”. In: *Networks* 49.4, pp. 258–272. DOI: 10.1002/net.20177.