This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

SHARP: A Scalable Framework for Dynamic Joint Replica Placement and Request Routing Scheduling

Yi Wang[1] Chen Tian[1] Hongbo Jiang[1] Xue Liu[2] Jinhua Chen[1] Wenyu Liu[1]
[1]Department of EIE, Huazhong University of Science and Technology, Wuhan, Hubei, China
[2]School of Computer Science, McGill University, Montreal, Quebec, Canada
[1]{ywang,tianchen,hxj,liuwy}@mail.hust.edu.cn, [1]chenjinhua.87@gmail.com, [2]xueliu@cs.mcgill.ca

Abstract—**This paper presents SHARP: a scalable framework for Dynamic Joint Replica Placement and Request Routing (DJRPRR) scheduling in content delivery networks. After grouping similar proxies and modeling them by a single section, we propose a hierarchical scheduling framework to greatly reduce the dimensions of the mathematical formulation. In every phase the obtained shaped formulation has an easy-solvable form and the complete optimization process is highly scalable. To verify the scalability and effectiveness of our approach, SHARP is evaluated by comprehensive experiment settings which are derived from realistic data/topology of an operational commercial CDN.**

I. Introduction

Content Delivery Networks (CDNs) can improve web content performance by delivering and servicing contents on an Internet Content Provider (ICP)'s behalf to end users: numerous proxies are placed at strategic network locations on the Internet to make the replicas of contents as close to users as possible. Because not all contents are available from all proxies, and not all proxies are operational at all times, the CDN provider needs to address two problems: how many replicas of each content should be placed and where to place them, which is the Replica Placement problem; how to choose the right proxies when requests for this content come, which is the Request Routing problem. These two problems are coupled and should be jointly scheduled.

Due to the nature of dynamic operations, the optimization of DJRPRR scheduling for an operational commercial CDN should be completed in a relatively short time. Existing methods model each proxy alone and solve this problem using Mixed Integer Programming. However, a large scale DJRPRR instance can not be solved in an acceptable short time to meet the dynamic requirement.

Instead, a scalable framework SHARP, which means Scalable Hierarchical Architecture for Replica Placement, is proposed in this paper. By grouping similar proxies into sections, we propose this hierarchical scheduling architecture to greatly reduce the dimensions of the mathematical formulation.

Our work based on the fact that CDN providers typically group those similar content together such as channels.In the rest of this paper, we use the terms *channel* and *content* interchangeably for the sake of convenience. Due to large resource requirements of video sharing, each such a site can be classified as a dedicated content class and be scheduled alone. Funded by a leading CDN provider, this research is supposed to develop a scalable framework for DJRPRR scheduling of large scale CDN multimedia service.

The remainder of this paper is organized as follows. Related work is provided in Section II. We formulate DJRPRR problem and present its scalability challenge in Section III. In Section IV we propose a scalable framework SHARP for DJRPRR schedulingComputational results of the comprehensive experiments are given in Section V, followed by conclusions in Section VI.

II. Related work

Dynamic DJRPRR scheduling solution is not provided adequately by previous researches. Most of previous works in CDN operational research field [2] deal with network planning problems. A major limitation of all these solutions is that they neglect to consider the natural dynamics in user requests. When content popularity changes, a different plan might be less costly, while re-applying the algorithms from scratch is definitely impractical for an already-established running CDN.

The research closest to our work was presented by Bartolini [1] and Bektas [3]. Bartolini [1] proposes the idea of periodically making decisions on joint replica placement and request routing scheduling problem. In Bektas [3], each client is modeled and should be assigned to a proxy by optimizing scheduling formulation. Modeling each client dramatically increases problem size and is inferior to modeling user access nodes [1]. What's more, the two decomposition algorithms of this work for Integer Programming formulation are too time-consuming hence not practical for dynamic purpose in large scale networks.

III. Problem Definition

A. Formulation and Scalability Challenge

In this section, we discuss the constraints and objective and present the complete mathematical formulation for DJRPRR. A CDN node has many proxy servers and a high speed connection to Internet. Like the container of content, a proxy has a limited storage capacity And the storage limitation of a proxy can be measured by the number of replicas that it can store. A proxy also has an upper bound of service power which can be denoted by servicing bandwidth limitation. All proxies in the same node share the node link bandwidth. Usually representing a topological district of an ISP, a user access node is the ingress of user requests at the edge of the network: all

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

predicted traffic from each access node should be fulfilled, and QoS constraints (e.g. user perceived latency) should be guaranteed.

The objective function of DJRPRR scheduling is to minimize the operational cost of CDNs. All costs are related to contents: transfer and maintenance. The replica maintenance cost is used to model the proxy cost of hosting replicas, reconfiguring replicas and the network cost to keep them up to date.

We denote individual proxy as i, $i \in \{1, \cdots, I\}$, individual user access node as k, $k \in \{1, \cdots, K\}$, individual content as j, $j \in \{1, \cdots, J\}$. We denote individual node as n, $n \in \{1, \cdots, N\}$, and each proxy i must reside in a node n.

For each proxy i, let B_i denote the maximum number of replicas that proxy i can store. Servicing limitation can be denoted by a maximum service bandwidth as G_i. For each node n, let W_n denote the bandwidth limitation. And we denote the request traffic from user node k for content j in next time slot as P_{kj}. C_{ki} denotes the unit transfer cost between k and i, and C_{ij} denotes the cost of updating and maintaining a replica of j at i.

The following decision variables are defined:

- s_{ij} denotes if a replica of content j is scheduled to be stored in proxy i in next time slot.
- z_{kij} denotes the scheduled request traffic of stored content j should be served to user k by proxy i in next time slot.

Then this DJRPRR scheduling problem is formulated as a Mixed Integer Programming formulation as follows:

$$\begin{aligned}
\text{Minimize: } & \sum_k \sum_i \sum_j z_{kij} C_{ki} + \sum_i \sum_j s_{ij} C_{ij} \\
\text{Subject to: } & (a) \sum_i z_{kij} = P_{kj}, \forall k, j \\
& (b) \sum_k \sum_{i \in n} \sum_j z_{kij} \leq W_n, \forall n \\
& (c) \sum_j s_{ij} \leq B_i, \forall i \\
& (d) \sum_k \sum_j z_{kij} \leq G_i, \forall i \\
& (e) \sum_k z_{kij} \leq s_{ij} G_i, \forall i, j \\
& (f) z_{ijk} \geq 0, s_{ij} \in \{0, 1\}, \forall k, i, j.
\end{aligned} \tag{1}$$

Constraint set (a) denotes that requests of every content from each user node should be fulfilled. (b) states that in each node the total servicing bandwidth should not exceed the node's outlet bandwidth limitation. (c) states that the total number of stored replicas can not exceed proxy physical limitation. (d) represents the limitation of proxy servicing power. (e) indicates that the a specific content can be scheduled to serve only if it is also scheduled to be stored. (f) defines the variables' feasible range.

The scalability of optimization process is critical for DJRPRR scheduling, that is, the optimization procedure should be completed in an acceptable short time even for a large scale instance. MIP problems are proved to be NP-hard and their time-complexities are exponential to the number of variables [4].

IV. A Scalable Framework for Dynamic Scheduling

A. Motivation

The CDN multimedia service topology is in nature divisional: many proxies (almost identical) reside in the same node. In our opinion, modeling each proxy with its constraints and its topology information is unnecessary. Grouping similar proxies and modeling them by a single section can greatly reduce the dimension of the problem. However even after grouping, the derived MIP formulation may still have a considerable scale that can't be solved in a short time. The basic idea of formulation shaping is: we deliberately modify formulation structures and break the original optimization problem into consequent optimization phases; in each phase the obtained shaped formulation should have an easy-solvable form, which guarantees that the whole optimization process can be highly scalable.

B. SHARP Framework

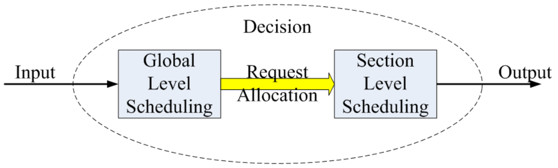

Fig. 1. Global Level and Section level Scheduling

SHARP consists of two levels: global level and section level. Global level is in charge of dynamic scheduling among network sections and user nodes: how many traffic of a specific content should be served to a specific user node by an individual section. Section level deals with scheduling inside each section. The design principle is: keep both optimization process and QoS constraints in global level; in section level user requests served by any proxy are supposed to meet the requirements of QoS; the section level scheduling should be formulated to a Generalized Assignment Problem [4].

Based on our design principle, the primary qualification of proxy grouping is: proxies are topologically adjacent and their QoS constraints are almost the same. Besides topological adjacency, proxy capability should also be taken into consideration. Proxies in the same section should have the same $\frac{G_i}{B_i}$ ratio, which implies same servicing power per replica. For example in a server node, we have

- proxy A with 1.5 TB storage size (can store one replica) and 200M bps servicing power;
- proxy B with 3 TB storage size (can store two replicas) and 400M bps servicing power.

They can be grouped into the same section with the standard of "200M bps per replica". After grouping, we normalize proxies to *slots*. In the same section each slot can store just one replica and has the same servicing power. Now a slot is the basic unit of scheduling. This normalization makes section level capable of scheduling a Generalized Assignment Problem form which is solvable in polynomial time [4].

C. Scheduling Formulation

We now present the formal optimization model of SHARP framework. New definitions and notations are given first. We denote individual section as m, $m \in \{1, \cdots, M\}$. We inherit i to present slot, and each slot i must reside in a section m. In the rest of the paper, we use the terms *slot* and *proxy* interchangeably.

For each section m, $B_m \in Z^+$ denotes the storage limitation (number of slots), and $W_m \in R^+$ denotes the bandwidth limitation of section m. We assume that a replica in a section m can support request traffic up to G_m bandwidth. C_{km} denotes the unit transfer cost between m and k, and C_{mj} denotes the cost of updating and maintaining a replica of j at m.

We define the following decision variables:

- s_{mj} denotes the scheduled number of replicas of content j in section m in next time slot;
- z_{kmj} denotes the scheduled request traffic of stored content j should be served to user k by section m in next time slot.

The complete global level scheduling formulation is as follows:

$$\begin{aligned} \text{Minimize: } & \sum_k \sum_m \sum_j z_{kmj} C_{km} + \sum_m \sum_j s_{mj} C_{mj} \\ \text{Subject to: } & (a) \sum_m z_{kmj} = P_{kj}, \forall k, j \\ & (b) \sum_j \sum_k z_{kmj} \leq W_m, \forall m \\ & (c) \sum_j s_{mj} \leq B_m, \forall m \\ & (d) \sum_k z_{kmj} \leq s_{mj} G_m, \forall m, j \\ & (e) z_{kmj} \geq 0, s_{mj} \in Z^+ \cup 0, \forall k, m, j. \end{aligned} \tag{2}$$

Constraint set (a) denotes that requests of every content from each user node should be fulfilled. (b) states that in each section the total servicing bandwidth should not exceed the section's bandwidth limitation. (c) states that the total number of stored replicas can not exceed section's physical limitation. (d) indicates that the traffic that can be provided to a specific content is limited by the number of stored replicas. (e) defines the variables' feasible range.

The QoS constraints are not included directly in the formulation because we can put them in the preprocessing procedure: each $\{k, m, j\}$ combination is examined if it is acceptable under QoS constraints. All violations can be treated as setting corresponding $z_{kmj} = 0$. Unlike incorporating QoS constraints in formulations [1], this preprocessing of QoS can be done in $O(kmj)$ complexity and greatly simplify the mathematical formulation. What's more, it reduces the number of variables of z_{kmj}. After global scheduling, the task of each section scheduling is to find a feasible solution under the constraints

$$\sum_{\forall i \in m} s_{ij} = s_{mj}, \forall j.$$

This is a Generalized Assignment Problem and can be solved in polynomial time [4]. In the rest of the paper, we focus on global level scheduling.

However the obtained global level formulation (2) is still an NP-hard MIP problem. It quickly becomes computationally intractable when the network scale further grows.

A common method for MIP problem is Lagrangean relaxation [5]. However this relaxation converges very slowly due to the integrity constraints of s_{mj} and can not meet the dynamic scheduling requirement.

D. Formulation Shaping

Abide by mathematical optimization principals, we first use *formulation shaping* to simplify the structures of the basic formulation (2) to an easy solvable form; secondly to accelerate the convergence of optimization process, the original optimization problem is broken into consequent optimization phases.

Due to its integrity constraint, the first step of formulation shaping is to remove s_{mj} as many as possible from the formulation. Here we denote the feasible region of the basic formulation (2) as $\mathcal{F}$. It is observed that the coefficients of s_{mj} are positive in the objective function, so s_{mj} should be minimized in the optimal solution. Then, we can remove constraint set (d) directly by defining

$$s_{mj} = \lceil \frac{\sum_k z_{mjk}}{G_m} \rceil, \forall m, j. \tag{3}$$

Then we dualize constraint set (c) to the objective function. Let $\{\lambda_m \geq 0, \forall m\}$ denote dual adjustable parameters, the relaxed objective function is

$$\sum_k \sum_m \sum_j z_{kmj} C_{km} + \sum_m \sum_j s_{mj} C_{mj} + \sum_m \lambda_m (\sum_j s_{mj} - B_m). \tag{4}$$

Noticed that

$$s_{mj} \geq \lceil \frac{\sum_k z_{mjk}}{G_m} \rceil, \forall m, j,$$

we can instead evaluate the lower bound of equation (4) (also of formulation (2)) by first substituting s_{mj} with $\frac{\sum_k z_{mjk}}{G_m}$ and then solve a new relaxation formulation:

$$\begin{aligned} \text{Minimize: } & \sum_k \sum_m \sum_j z_{kmj} (C_{km} + \frac{C_{mj} + \lambda_m}{G_m}) - \sum_m \lambda_m B_m \\ \text{Subject to: } & (a) \sum_m z_{kmj} = P_{kj}, \forall k, j \\ & (b) \sum_k \sum_j z_{kmj} \leq W_m, \forall m \\ & (e) z_{mjk} \geq 0, \forall k, m, j. \end{aligned} \tag{5}$$

The remaining structure of this relaxation is a standard LP problem which can be solved in polynomial time by interior point method [4]. Suppose that a solution of Lagrangean Dual (LD) formulation (5) can be found as (z^*_{kmj}, s^*_{mj}). If this solution satisfies both equation (3) and the constraints set (c) of formulation (2), then it is in $\mathcal{F}$. Lower Bound (LB) of the optimum objective function value can be given by objective value of formulation (5) and the Upper Bound (UB) of the basic formulation can be given as:

$$UB = \sum_k \sum_m \sum_j z^*_{kmj} C_{km} + \sum_m \sum_j s^*_{mj} C_{mj}.$$

Note that UB indicates the current best feasible solution. A solution is out of $\mathcal{F}$ if any constraint in dualized constraint set (c) is violated. If in each iteration the solution of the relaxed formulation (5) is always in $\mathcal{F}$, LD can converge quickly to the optimum (or near-optimum) objective value of the basic formulation (2). The second step of formulation shaping is to make the solution of formulation (5) always stay in $\mathcal{F}$.

Theorem: If $L_m \le (B_m - J)G_m, \forall m$ and we replace all W_m with L_m in formulation (5) as

$$(b) \sum_j \sum_k z_{kmj} \le L_m, \forall m,$$

then any feasible solution of formulation (5) is also feasible for formulation (2).

Proof:

$$\begin{aligned} s_{mj} &= \lceil \frac{\sum_k z_{kmj}}{G_m} \rceil \le \frac{\sum_k z_{kmj}}{G_m} + 1 \\ \Rightarrow \sum_j s_{mj} &\le \sum_j (\frac{\sum_k z_{kmj}}{G_m} + 1) \le \frac{L_m}{G_m} + J \\ \Rightarrow \sum_j s_{mj} &\le (B_m - J) + J = B_m. \end{aligned}$$

So if we set $L_m = min\{W_m, (B_m - J)G_m, \forall m\}$, and the dual constraint set (c) should be satisfied for all solutions of the shaped formulation.

One may question that if

$$\sum_k \sum_j P_{kj} \le \sum_m W_m$$
$$\sum_k \sum_j P_{kj} > \sum_m L_m,$$

the basic formulation may be feasible while the shaped formulation is infeasible. And even if

$$\sum_k \sum_j P_{kj} \le \sum_m L_m \qquad (6)$$

holds but if

$$W_m \ge (B_m - J)G_m, \exists m,$$

the shaping of section bandwidth reduces the feasible region $\mathcal{F}'$ of the shaped formulation(compared with $\mathcal{F}$). This situation may lead to deviating from the optimum value.

E. Two Phase Global Scheduling

It is observed that the lower J value, the higher L_m and hence the closer $\mathcal{F}'$ to $\mathcal{F}$. If the number of contents in the formulation can be tailed, the formulation can be shaped to preferable form. We partition the global level scheduling to two phases: primary round scheduling and remainder scheduling, as shown in Fig. 2. We select the majority of contents into primary round to guarantee that equation (6) holds. Intuitively, the traffic of contents in the primary round should be maximized and those contents with little traffic should be removed first.

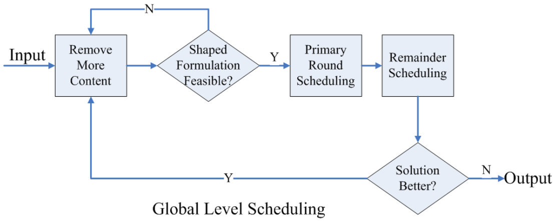

Fig. 2. Global scheduling with content selection

The remaining small subset of contents is scheduled in remainder scheduling phase. As long as the basic formulation is feasible, there should be bandwidth and proxy not allocated after the primary round. All these redundant bandwidth and proxies can be allocated to unprocessed contents in the second phase. It is observed that remainder scheduling can be formulated again as the same form of formulation (2), except in a much smaller scale. After the primary round, the remaining traffic in remainder scheduling phase should be relatively small. We can iteratively apply all aforementioned steps to optimize it. Or, instead of precisely optimization, simple greedy heuristics may be acceptable for this phase.

V. Evaluation

A. Setup

As traffic prediction is not one of our keystones, we develop a traffic generator to simulate the traffic distribution predictions as the inputs of experiments. This generator is driven by realistic trace data of a leading video site in China.

We obtained system logs and collected the traffic records of 41 video channels for a week from Jan 8 to Jan 14, 2008 in Shenyang section. In our experiments, we first obtain sample distributions on the actual data. For each content in a simulation, we use a randomly generated value following the distribution. After randomization, all content traffic is scaled to meet the requirement of total aggregated traffic in a specific experiment.

We use the interior point algorithm in GLPK software package [8] to solve large scale LP subproblems. The Lagrangean relaxation is solved by standard sub-gradient method [6]. We set initially $\lambda_m = 0$ and adjust step length to 1. We here set the converge threshold of shaped formulation as the ratio of LB/UB(eg. 0.9). All computation experiments are performed on an Intel Q6600 computer with 3GB RAM.

To simplify simulation setups, we set all proxies to have 1.5TB size and 200MB power each. Similarly we set the same number of proxies in every section. All other system settings and algorithm parameters are manageable, including user nodes number, section number, contents number, proxies per section, LB/UB ratio threshold etc.

We study SHARP performance by a wide spectrum of system configurations. The experiments are performed in the most demanding scheduling conditions. We suppose $W_m \ge (B_m - J)G_m, \forall m$ and L_m is calculated by $(B_m - J)G_m$. C_{mk} are randomly chosen from 2 to 11; the same is C_{mj}, ranging from 2 to 4. Each data is the mean of 100 independent experiments with C_{mk} and C_{mj} randomization.

For performance comparison, the most important metric is computational time (in seconds). As the randomization of request traffic plays a key role in objective values, the gap between the solution value and the lower bound of optimum value is chosen as the optimum approximation indicator.

As mentioned above, there may be a gap between shaped formulation and the optimum solution of the basic formulation. As formulation (2) is not applicable for large dynamic instances, first we test the performance of both formulations in two relatively small scale experiments. A simple greedy heuristic is chosen for remainder scheduling phase. Convergence condition of shaped formulation primary round is set to $LB/UB \geq 0.99$. Implemented by SYMPHONY [7], we set the convergence condition of basic formulation also to $LB/UB \geq 0.99$.

B. Abundant Resource Case Study

The experiment topology is: 10 user nodes, 20 sections and 41 contents. The traffic is directly generated by amplifying 41 channels trace data of Saturday by 4 times. The total aggregated traffic is 350,968M bps and at least $350,968/200 \approx 1,755$ proxies are needed. Each section has 128 proxies and a bandwidth of 19,200M bps. The bandwidth limitation is relatively tight as $350,968/(19,200 \times 20) \approx 91.4\%$; and the proxy resources may be abundant as $1,755/(128 \times 20) \approx 68.6\%$.

Let R denote the number of removed contents, we evaluate R from 1 to 8. We refer O_b and O_s as the obtained objective values of basic formulation and shaped formulation respectively, T_b and T_s as total computational time consumed of SYMPHONY and shaped formulation respectively.

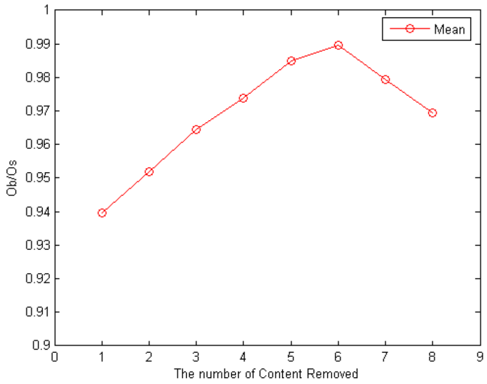

Fig. 3. Solution Approximation

As shown in Fig. 3, SYMPHONY does get better solution. This gap between SYMPHONY and shaped formulation is because of the heuristics existing in formulation shaping. However the value gap is small: even in the worst case the approximation ratio is over 94%. The value of R does have effect on solution approximation. If $R = 1$, $L_m = (128 - 40) \times 200 = 17,600$. The total bandwidth provided to primary round is $17,600 \times 20 = 35,200$, just a litter higher than total traffic in primary scheduling. The feasible region of shaped formulation is too small and the optimum approximation is also limited. Intuitively the bigger R value is, the more traffic removed from primary round schedule and the more section bandwidth provided for primary round. In this scenario, $R = 6$ gets the best solution of global scheduling. After that, the approximation of our SHARP schedule declines. We find the reason by trace and analysis. As L_m is already big enough, scheduling can't get more benefit from allocating more bandwidth to primary round; at the same time when R increases, more traffic is left to be scheduled by greedy algorithm in second phase. Compared with optimization in primary round, heuristics are inferior in performance.

TABLE I
Optimum Verification

	T_b	$T_s(4)$	$T_s(5)$	$T_s(6)$	$T_s(7)$	$T_s(8)$
time(s)	710.8	≤ 1	≤ 1	≤ 1	≤ 1	≤ 1

The time consumption is shown in Table I. $T_s(R)$ denotes that there are R contents removed from primary phase. Shaped formulation exhibits superior performance: all shaped formulation optimizations together with second phase greedy algorithms need less than 1 second. As a comparison, the basic formulation needs over 700 seconds to be optimized.

VI. Conclusion

This paper presents a scalable framework for Dynamic Joint Replica Placement and Request Routing scheduling in content delivery networks. Most previous operational research in CDN focusing on network planning problems neglects to consider the natural dynamics in the user requests traffic. As a result, existing methods lack scalability and incur high time complexity. And our SHARP approach, which is based on a hierarchical scheduling architecture and formulation shaping, outperforms them both in scalability and efficiency.

Acknowledgment

This work was supported in part through National Natural Science Foundation of China (No.60803115, No. 60873127, No.61073147), National Natural Science Foundation of China - Microsoft Research Asia (No.60933012), the Youth Chenguang Project of Wuhan City (No.201050231080), the Scientific Research Foundation for the Returned Overseas Chinese Scholars (State Education Ministry), and the Program for New Century Excellent Talents in University (State Education Ministry, No. NCET-10-408).

References

[1] Novella B, Francesco LP and Chiara P, Dynamic Replica Placement and User Request Redirection in Content Delivery Networks, ICC 2005.

[2] Bektas T, Oguz O, Ouveysi I. Designing cost-effective content distribution networks. Computers & Operations Research 2007;34:2436-49.

[3] Bektas T, Cordeau JF, ErkutE and Laporte G.Exact algorithms for the joint object placement and request routing problem in content distribution networks. Computers & Operations Research (doi:10.1016/j.cor.2007.02.005) (In Press)

[4] G.L. Nemhauser and L.A. Wolsey. Integer and Combinatorial Optimization. New York: John Wiley, 1988.

[5] M. L. Fisher. The Lagrangian Relaxation Method for Solving Integer Programming Problems. Management Science, 27(1):1-18, 1981.

[6] Bazaraa, M.S. and Sherali, H.D.: On the choice of step size in subgradient optimization. European Journal of Operational Research, 7, 1981, pp. 380-388.

[7] SYMPHONY Project, URL:https://projects.coin-or.org/SYMPHONY

[8] GLPK-GNU Project, URL:http://www.gnu.org/software/glpk/