ICIC Express Letters ICIC International ©2017 ISSN 1881-803X
Volume **11**, Number **12**, December **2017** pp. 1757–1763

AN EXACT ALGORITHM FOR THE ROBUST BANDWIDTH PACKING PROBLEM

Jinil Han

Department of Industrial and Information Systems Engineering
Soongsil University
369 Sangdo-ro, Dongjak-gu, Seoul 06978, Korea
jinil.han@ssu.ac.kr

Received June 2017; accepted September 2017

Abstract. *The bandwidth packing problem (BWP) that arises in the area of telecommunication is considered. We additionally address demand uncertainty in the BWP. In order to produce qualified solutions under such an uncertainty, we apply robust optimization technique. The resulting robust BWP becomes difficult integer program. Thus, we propose an efficient algorithm based on the Dantzig-Wolfe reformulation of the robust BWP. Computational results show that our algorithm outperforms commercial solvers.*
Keywords: Bandwidth packing, Robust optimization, Exact algorithm, Branch-and-price

1. **Introduction.** Bandwidth packing problems commonly arise in the area of telecommunication networks. When calls are given along a fixed bandwidth between a pair of nodes, the goal is to determine the selection of calls and the assignment of a single path to each selected call such that edge capacities are not exceeded and profit from selected calls is maximized. The single path requirement makes the BWP NP-hard. Various solution approaches have been considered to solve the BWPs, including tabu search, Lagrangian relaxation, and column generation [1-4]. Some works have also investigated additional issues, such as multiple periods [5], queueing delay [6], and priority classes [7], to name a few.

In this paper, we address demand uncertainty in the classical BWP. The amount of bandwidth used for each call is typically not fixed and rather changes over time. Moreover, it is difficult or even impossible to estimate bandwidth reliably in large networks. In this case, the solution of deterministic problem that uses mean demand may cause in a deterioration in the quality of service, and therefore a network manager tends to estimate the demand conservatively to safely guarantee the feasibility of selected calls. Then, it obviously leads to a wastage of network capacities. To efficiently handle this data uncertainty, we use a robust optimization technique in the sense of Bertsimas and Sim [8].

Robust optimization is a methodology for handling optimization problem with uncertain data, which aims at finding solutions that are feasible for all realization of data in given uncertainty set and optimize against the worst-case instance. At a first glance, robust optimization seems a very conservative approach. However, it can provide a parameter that enables us to control the level of conservatism of the solution. Moreover, the model proposed by Bertsimas and Sim [8] has the advantage of preserving the linearity of the original problem, and therefore the robust problem of an integer program is also an integer program. However, for large-scale integer program, its robust problem may not be solved efficiently using MIP solvers. Thus, we develop an efficient algorithm based on Dantzig-Wolfe reformulation to solve the robust bandwidth packing problem (RBWP).

The paper is structured as follows. Section 2 describes the problem formulations for the RBWP and our Dantzig-Wolfe reformulation is given in Section 3. The detailed branch-and-price algorithm is given in Section 4. Computational test results are reported in Section 5. Finally, the concluding remarks are given in Section 6.

2. **Problem Formulation.** In this section, we present an integer programming formulation for the RBWP. Let $G = (V, E)$ be an undirected graph where V is the set of nodes and E is the set of edges. Let $A(v)$ be the set of adjacent nodes of node $v \in V$. Let K denote the set of calls, i.e., the set of origin-destination pairs with demand requirement. The origin of call $k \in K$ is s_k and its destination is t_k. Let b_e denote the capacity of $e \in E$ and w_k denote the revenue of call $k \in K$. We assume the demand of a call k is distributed with mean $\bar{r}_k > 0$ and maximum deviation $\hat{r}_k > 0$. In other words, the actual demand can take values in $[\bar{r}_k - \hat{r}_k, \bar{r}_k + \hat{r}_k]$ for every call $k \in K$. Given this demand uncertainty set, the RBWP is to find a maximum revenue selection of calls while routings of the calls observe the capacity of the links if at most Γ calls are allowed to deviate from their mean value simultaneously. We can control the level of conservatism of the solution by controlling parameter Γ. Note that when $\Gamma = 0$, the robust problem is equivalent to the deterministic BWP. Likewise, if $\Gamma = |K|$, we find the solution against the worst-case demand realization. Therefore, by varying $\Gamma \in [0, |K|]$, we have the flexibility of adjusting the robustness against the level of conservatism of the solution. Before presenting the RBWP formulation, we give the following deterministic BWP, where the nominal demand equals its mean value.

$$\textbf{(RBWP)} \quad \max \sum_{k \in K} w_k y_k \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in A(v)} x_{ji}^k - \sum_{j \in A(v)} x_{ij}^k = \begin{cases} -y_k, & \text{if } i = s_k; \\ y_k, & \text{if } i = t_k; \\ 0, & \text{otherwise} \end{cases} \quad \forall k \in K,\ i \in V, \tag{2}$$

$$x_{ij}^k + x_{ji}^k \le x_e^k,\ \forall k \in K,\ e = \{i, j\} \in E, \tag{3}$$

$$\sum_{k \in K} \bar{r}_k x_e^k + \max_{\{S \subseteq K, |S| \le \Gamma\}} \sum_{k \in S} \hat{r}_k x_e^k \le b_e,\ \forall e \in E, \tag{4}$$

$$x_{ij}^k \in \{0, 1\},\ y_k \in \{0, 1\}, \tag{5}$$

where y_k takes value 1 if call k is selected, x_{ij}^k takes value 1 if call k is routed through a path that uses arc ij. Constraints (2) contain the flow conservation equations, and arc variable x_{ij}^k and edge variable x_e^k are linked together in constraints (3). Constraints (4) ensure that the total demand routed through each edge cannot exceed the edge capacity, where an inner maximization problem determines the maximum total deviation from the mean demand values when at most Γ calls can be deviated simultaneously. Note that this problem is a nonlinear program; however, by taking dual of the inner maximization problem we can reformulate it as a following problem.

$$\begin{aligned} (\mathbf{P_{BS}}) \quad \max \quad & (1) \\ \text{s.t.} \quad & (2), (3), (5), \\ & \sum_{k \in K} \bar{r}_k x_e^k + z_e \Gamma + \sum_{k \in K} p_e^k \le b_e,\ \forall e \in E, \\ & z_e + p_e^k \ge \hat{r}_k x_e^k,\ \forall e \in E,\ \forall k \in K, \\ & p_e^k \ge 0,\ \forall e \in E,\ \forall k \in K, \\ & z_e \ge 0,\ \forall e \in E. \end{aligned}$$

For a detailed explanation of this reformulation, refer to [8]. This reformulation enables us to solve the RBWP using MIP solvers since it preserves the linearity of the deterministic problem ($\Gamma = 0$). Note that, compared to the deterministic problem, we need $|E| + |E||K|$ additional variables and $|E||K|$ additional constraints for the robust problem. Accordingly, if the given network size and the number of calls are large, solving this reformulation directly using the solvers becomes difficult. Therefore, we apply the Dantzig-Wolfe decomposition and present the branch-and-price algorithm to find optimal solution of the RBWP.

3. **Dantzig-Wolfe Reformulation.** To develop the Danzig-Wolfe reformulation, we first define a call pattern, which represents the set of possible calls that can traverse the same edge simultaneously [4,6]. A call pattern g for edge e is mathematically represented by a set $K(g)$, such that $\sum_{k\in K(g)} \bar{r}_k \leq b_e$. This definition of call pattern can be extended to the context of robust bandwidth packing, such that a Γ-robust call pattern for edge e is defined as a binary integer solution of the knapsack constraint, $\sum_{k\in K} \bar{r}_k x_k + \max_{\{S\subseteq K,|S|\leq\Gamma\}} \sum_{k\in S} \hat{r}_k x_k \leq b_e$.

Let $P(k)$ be the set of (s_k, t_k)-paths of call $k \in K$ and $P(k,e)$ be the set of paths in $P(k)$ that traverse edge $e \in E$ for call $k \in K$. Let $G_\Gamma(e)$ denote the set of all Γ-robust call patterns for edge $e \in E$ and $G_\Gamma(e,k)$ denote the set of Γ-robust call patterns of $e \in E$ containing call $k \in K$. Then, the following Dantzig-Wolfe reformulation can be developed.

$$
\begin{aligned}
(\mathbf{P_{DW}}) \quad \max \quad & \sum_{k\in K}\sum_{p\in P(k)} w_k y_k^p \\
\text{s.t.} \quad & \sum_{p\in P(k)} y_k^p \leq 1, \ \forall k \in K, && (6)\\
& \sum_{g\in G_\Gamma(e)} z_e^g \leq 1, \ \forall e \in E, && (7)\\
& \sum_{p\in P(k,e)} y_k^p \leq \sum_{g\in G_\Gamma(e,k)} z_e^g, \ \forall e \in E, \ k \in K, && (8)\\
& y_k^p \in \{0,1\}, \ z_e^g \in \{0,1\},
\end{aligned}
$$

where y_k^p takes value 1 if call k is selected and path p is assigned to call k, 0 otherwise, and z_e^g takes value 1 if call pattern g of edge e is selected, 0 otherwise. Constraints (6) mean that a maximum of one path can be selected for each call. Constraints (7) ensure that at most one Γ-robust call pattern can be assigned to each edge. Constraints (8) mean that a path for call k can traverse edge e only if the call pattern containing call k is selected for edge e. Since the formulation $\mathbf{P_{DW}}$ has exponentially many variables, it is not possible to solve it using MIP solvers. However, the LP relaxation of $\mathbf{P_{DW}}$ is suitable for applying column generation. The details of our algorithm including the column generation are presented in the next section.

4. **Solution Method.** We now propose a branch-and-price approach to solve our reformulation $\mathbf{P_{DW}}$. It combines column generation with a branch-and-bound to obtain integer solutions. For more details on the general methodology of the branch-and-price algorithm, the reader is referred to [9]. Note also that our branch-and-price algorithm is similar to that of Han et al. [6] proposed to solve a class of nonlinear BWP, except that they solve nonlinear knapsack problems as a subproblem. For more details, see [6].

The idea of column generation for solving the LP relaxation is as follows. Let $\hat{P}(k)$ be the set of paths of call k generated so far and $\hat{G}_\Gamma(e)$ be the set of Γ-robust call patterns of edge e generated so far. The restricted master problem $\mathbf{RM_{DW}}$ of $\mathbf{P_{DW}}$ can be obtained from $\mathbf{P_{DW}}$ by replacing $P(k)$ and $G_\Gamma(e)$ with $\hat{P}(k)$ and $\hat{G}_\Gamma(e)$, respectively. Let α_k, β_e

and γ_e^k be dual variables associated with constraints (6), (7) and (8), respectively. We can solve the LP relaxation of $\mathbf{P_{DW}}$ by solving several $\mathbf{RM_{DW}}$s repeatedly. After finding an optimal solution to $\mathbf{RM_{DW}}$, we check whether there are any columns not included in the $\mathbf{RM_{DW}}$ with a positive reduced cost. If such columns exist, they are added to the $\mathbf{RM_{DW}}$, and the process is repeated; otherwise, the current optimal solution to $\mathbf{RM_{DW}}$ is also optimal to the LP relaxation of $\mathbf{P_{DW}}$.

Let $\left(\bar{\alpha}_k, \bar{\beta}_e, \bar{\gamma}_e^k\right)$ be the optimal dual values for the current $\mathbf{RM_{DW}}$. For each iteration, we generate two types of columns, i.e., path and pattern. First, the reduced cost of path p for call k is: $w_k - \bar{\alpha}_k - \sum_{e\in E(p)} \bar{\gamma}_e^k$, where $E(p)$ is the set of edges contained in the path p. Then we solve the following subproblem to find paths with a positive reduced cost for each call $k \in K$.

$$
\begin{aligned}
(\mathbf{SP1}(k)) \quad \min \quad & \sum_{e\in E(p)} \bar{\gamma}_e^k \\
\text{s.t.} \quad & p \in P(k).
\end{aligned}
$$

For each call $k \in K$, $\mathbf{SP1}(k)$ is the shortest path problem from the source node of call k to the sink node of call k over a network with positive edge costs; therefore, it can be efficiently solved by Dijkstra's algorithm. If the optimal cost is less than $w_k - \bar{\alpha}_k$, a new path for call k can be added to $\mathbf{RM_{DW}}$.

Next, the reduced cost of pattern g for edge e is: $-\bar{\beta}_e + \sum_{k\in K(g)} \bar{\gamma}_e^k$. Here, $K(g)$ is the set of calls contained in pattern g. We have to solve the following second subproblem to find call patterns with a positive reduced cost for each edge $e \in E$.

$$
\begin{aligned}
(\mathbf{SP2}(e)) \quad \max \quad & \sum_{k\in K} \bar{\gamma}_e^k x_k \\
\text{s.t.} \quad & \sum_{k\in K} \bar{r}_k x_k + \max_{\{S\subseteq K, |S|\le\Gamma\}} \sum_{k\in S} \hat{r}_k x_k \le b, \\
& x_k \in \{0,1\}, \ k \in N.
\end{aligned}
$$

For each edge $e \in E$, $\mathbf{SP2}(e)$ is a robust knapsack problem, which is NP-hard since the ordinary binary knapsack problem is a special case if $\Gamma = 0$. If the optimal cost is greater than $\bar{\beta}_e$, a new call pattern for edge e can be added to $\mathbf{RM_{DW}}$. Note that when we apply column generation, the demand uncertainty of the original problem is transferred to the robust knapsack subproblem and we have to solve $\mathbf{SP2}(e)$ repeatedly, which seems to be somewhat inefficient. However, the following proposition shows that it can be solved by solving the ordinary knapsack problems several times. Since the ordinary knapsack problem can be solved fast using well-known dynamic programming, it enables the robust knapsack subproblem not to be a bottleneck of the overall algorithm.

Proposition 4.1. [10] *The robust knapsack problem* ($\mathbf{SP2}(e)$) *can be solved by solving at most* $n - \Gamma + 1$ *ordinary knapsack problems.*

5. **Computational Results.** The algorithm was coded in C# and all tests were performed on an AMD X4 3GHz PC with 4GB RAM. ILOG CPLEX 12.1 was used for solving mixed integer program $\mathbf{P_{BS}}$ and master linear programs for the branch-and-price algorithm. To solve knapsack problems, *minknap* algorithm of Pisinger [11] was used.

We performed several computational experiments on a 10-nodes network that were used by Klopfenstein and Nace [12] in their work, except that our networks are undirected. We present pictures of the network in Figure 1(a). Every edge e has the same capacity $b_e = 100$. We randomly generated 40 calls, i.e., a source node and a sink node for each call are determined randomly. The nominal demand value of each call is integer randomly chosen in the range of $[20, 40]$. The demand is subject to uncertainty: for each call k, we

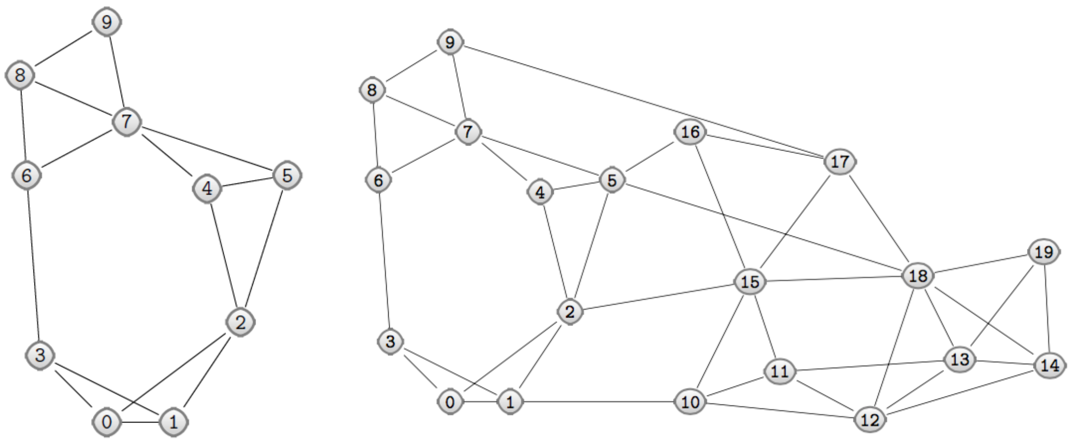

FIGURE 1. Test networks: (a) network 1; (b) network 2

set $\hat{r}_k = \alpha\bar{r}_k$, where $\alpha \in (0, 1]$. Finally, the revenue of each call is integer randomly chosen in the range of $[10, 20]$.

Table 1 shows the results. The meanings of the headings are summarized as follows:

#node: the number of generated nodes in the branch-and-bound tree.

#path: the total number of generated paths.

#ptn: the total number of generated patterns.

#lp: the total number of master LP solved.

root: the optimal value at the root node of the branch-and-bound tree.

opt: the integer optimal value (or the value of the best integer solution if no optimal solution is found within time limit)

gap: (root − opt) × 100/(opt).

lp: the total time spent in solving the master LPs in seconds.

sp: the total time spent in solving the shortest path problem in seconds.

kp: the total time spent in solving the knapsack problem in seconds.

hr: the total time spent in primal heuristic in seconds.

total: the total time spent in solving the problem in seconds.

TABLE 1. Results on Network 1 with $\alpha = 30\%$

Γ	algorithm	#node	#path	#ptn	root	opt	gap	lp	sp	kp	hr	total
0	bnp	372	433	1371	335.8	325	3.34	8.7	0.5	0.6	1.0	12.2
	cplex	437			342.5	325	5.40					4.2
1	bnp	9973	602	2179	310.6	292	6.39	275.5	4.7	36.7	9.2	345.1
	cplex	140828			325.3	292	11.42					712.5
2	bnp	1686	491	1274	283.5	269	5.41	24.1	1.0	6.5	2.0	37.9
	cplex	171101			312.8	268	16.30					5.14*
3	bnp	540	434	1024	274.4	262	4.77	5.9	0.5	2.5	0.6	11
	cplex	268501			302.3	260	15.40					2.86*

A total solution time limit is fixed at 1800 seconds. When this limit is reached, the total time in the table is expressed as * and the closed gap, which is the gap between the best integer objective and the objective of the best node remaining, is indicated. Note that $\Gamma = 0$ corresponds to the deterministic BWP, and the result for $\Gamma \geq 4$ is not reported, since $\Gamma = 4$ would already corresponds to the worst case. For each Γ value, the first row reports the performance of the branch-and-price. Also, the second row reports the result of CPLEX to solve $\mathbf{P_{BS}}$. We used the default CPLEX parameters.

When $\Gamma = 0$, the deterministic problems were solved very fast by CPLEX. However, for the robust problems with $\Gamma > 0$, our branch-and-price algorithm significantly outperforms the CPLEX. We can observe that all the tested instances were solved in the given time limit, while CPLEX was unable to find optimal solutions for the instances with $\Gamma > 2$. The root gaps for $\mathbf{P_{BS}}$ are very large compared to the Dantzig-Wolfe reformulations. The results for solving time indicate that the robust knapsack problem was solved very fast. This is because the robust knapsack problem is solved by customized dynamic programming algorithm [11] independently of CPLEX. Thus, we can say that the robustness of the original problem was well transformed to the column generation subproblems, which gives us the significant computational benefit.

We next tested our algorithm for relatively bigger instance presented in Figure 1(b). All data were generated in the same manner as the previous network, except that the number of calls increased to 80.

Table 2 provides the results on Network 2. Due to the large size of the instance, CPLEX was unable to solve even the deterministic problem and gives very poor solutions for robust problem with $\Gamma \geq 2$, such that closed gaps remain very large, up to 51% for the instance with $\Gamma = 3$. However, the branch-and-price algorithm provides the solutions close to being optimal for all Γ values, such that closed gaps are about 1%.

TABLE 2. Results on Network 2 with $\alpha = 30\%$

Γ	algorithm	#node	#path	#ptn	root	opt	gap	total
0	bnp	31828	1993	1410	1261.08	1260	0.00	0.08*
	cplex	2027			1261.50	1260	0.00	0.08*
1	bnp	34459	2338	1550	1243.55	1229	0.01	1.13*
	cplex	1163			1256.85	1228	0.02	2.23*
2	bnp	35106	2050	1249	1224.05	1206	0.01	1.41*
	cplex	809			1252.00	896	0.04	39.64*
3	bnp	24451	2347	1572	1204.78	1190	0.01	1.18*
	cplex	735			1246.89	824	0.05	51.17*

Next, to assess the robustness of the solution, we designed the simulation tests. Each demand value is assumed to follow a normal distribution with mean of $\bar{r}_k$ and a standard deviation of $\hat{r}_k/2$. Note that the range $[\bar{r}_k - 0.3\bar{r}_k, \bar{r}_k + 0.3\bar{r}_k]$ can be regarded as the confidence intervals of approximately 95% for this normal distribution. Then we generated 1000 demand scenarios and checked the feasibility of the solutions for each demand scenario. For the Network 1, the results are shown in Figure 2. The percentage of demand scenarios that cannot be flowed feasibly is reported for the varying Γ values. It

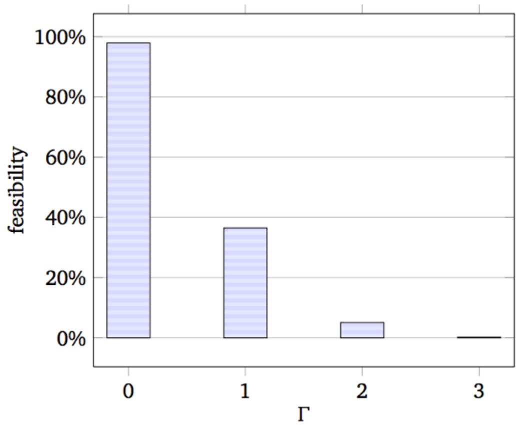

FIGURE 2. Percentage of infeasible scenarios

shows that 97.9% of all demand scenarios failed to flow all demands for the deterministic solution, while the robust solutions greatly improve the robustness with the increase of the parameter Γ. We can also observe that it is sufficient to set $\Gamma = 2$ to obtain solution which is robust with sufficiently high probability of feasibility.

6. **Conclusion.** In this paper, we present the Dantzig-Wolfe reformulation using Γ-robust call pattern for the robust bandwidth packing problem under the framework of [8] and propose the branch-and-price algorithm to solve it. In this approach, the demand uncertainty in the original problem is transferred to the robust knapsack problem that can be efficiently solved by applying the dynamic programming algorithm for ordinary knapsack problem several times, and thus it enables us to obtain computational gain. The computational results also demonstrate that our algorithm outperforms the CPLEX which solves the direct reformulation of [8]. We hope that our work contributes to the researches on the application of the general model of robust optimization to the large-scale mixed integer programming problems. Future research may include the investigation of probabilistic approach to the BWP under uncertainty. Note that probabilistic approach usually requires solving stochastic programming problems that are known to be much harder than robust optimization problems.

Acknowledgment. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2017R1C1B2006144).

REFERENCES

[1] B. Gavish and K. Altinkemer, Backbone network design tools with economic tradeoffs, *Informs Journal on Computing*, vol.2, no.3, pp.236-252, 1990.
[2] M. Parker and J. Ryan, A column generation algorithm for bandwidth packing, *Telecommunication Systems*, vol.2, no.1, pp.185-195, 1993.
[3] M. Laguna and F. Glover, Bandwidth packing: A tabu search approach, *Management Science*, vol.39, no.4, pp.492-500, 1993.
[4] K. Park, S. Kang and S. Park, An integer programming approach to the bandwidth packing problem, *Management Science*, vol.42, no.9, pp.1277-1291, 1996.
[5] A. Amiri and R. Barkhi, The multi-hour bandwidth packing problem, *Computers and Operations Research*, vol.27, no.1, pp.1-14, 2000.
[6] J. Han, K. Lee, C. Lee and S. Park, Exact algorithms for a bandwidth packing problem with queueing delay guarantees, *Informs Journal on Computing*, vol.25, no.3, pp.585-596, 2013.
[7] I. Bose, Bandwidth packing with priority classes, *European Journal of Operational Research*, vol.192, no.1, pp.313-325, 2009.
[8] D. Bertsimas and M. Sim, The price of robustness, *Operations Research*, vol.52, no.1, pp.35-53, 2004.
[9] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh and P. Vance, Branch-and-price: Column generation for solving huge integer programs, *Operations Research*, vol.46, no.3, pp.316-329, 1998.
[10] C. Lee, K. Lee, K. Park and S. Park, Branch-and-price-and-cut approach to the robust network design problem without flow bifurcations, *Operations Research*, vol.60, no.3, pp.604-610, 2012.
[11] D. Pisinger, A minimal algorithm for the 0-1 knapsack problem, *Operations Research*, vol.45, no.5, pp.758-767, 1997.
[12] O. Klopfenstein and D. Nace, Cover inequalities for robust knapsack sets – Application to the robust bandwidth packing problem, *Networks*, vol.59, no.1, pp.59-72, 2012.