Spectrum Bidding in Wireless Networks and Related

Xiang-Yang Li*[1], Ping Xu[1], ShaoJie Tang[1], and XiaoWen Chu[2]

[1] Illinois Institute of Technology, Chicago, IL, USA,
xli@cs.iit.edu, pxu3@iit.edu, stang7@iit.edu
[2] Hong Kong Baptist University, Hong Kong, China, chxw@comp.hkbu.edu.hk

Abstract. In this paper, we study the spectrum assignment problem for wireless access networks. Opportunistic spectrum usage is a promising technology. However, it could suffer from the selfish behavior of secondary users. In order to improve opportunistic spectrum usage, we propose to combine the game theory with wireless modeling. Several versions of problems are formalized under different assumptions. We design PTAS or efficient approximation algorithms for each of these problems such that overall social benefit is maximized. Finally, we show how to design a truthful mechanism based on all these algorithms.

1 Introduction

Wireless technology is expected to play a bigger and more fundamental role in the new Internet than it has today. The radio frequency spectrum has been chronically regulated with static spectrum allocation policies since the early $20th$ century. With the recent fast growing spectrum-based services and devices, remaining spectrum available for future wireless services is being exhausted, known as the *spectrum scarcity problem*. Current fixed spectrum allocation scheme leads to significant spectrum *white spaces* where many allocated spectrum blocks are used only in certain geographical areas and/or in brief periods of time. A huge amount of precious spectrum (below 5GHz), perfect for wireless communications, sit there silently. Recognizing that the traditional spectrum management process can stifle innovation, and it is difficult to provide a certain quality of service (QoS) for systems operated in unlicensed spectrum, the FCC has proposed several new spectrum management models [15].

One promising technology is the opportunistic spectrum usage, secondary users observe channel availability dynamically and explore it opportunistically. While opportunistic spectrum has several advantages, it suffers from selfish behavior of secondary users. Thus, we propose to combine the game theory [13] with

* The research of the author was partially supported by National Basic Research Program of China (973 Program) under grant No. 2006CB30300, the National High Technology Research and Development Program of China (863 Program) under grant No. 2007AA01Z180, the RGC under Grant HKBU 2104/06E and CERG under Grant PolyU-5232/07E. Part of the work was done when the author visited MSRA.

wireless communication modeling. More specifically, we study how to share the spectrum and how to charge secondary users such that the *overall social benefit* is maximized even in the presence of selfish behavior, while each secondary user specifies channel, space and time constraints. Note the correlation of time and space constraints introduces high complexity compared with traditional auctions [12, 16, 7, 6, 14, 2].

The main contributions of this paper are as follows. First we design efficient algorithms to allocate channels such that the social efficiency are approximately maximized. Based on these approximation algorithms, we then design strategyproof mechanisms to charge the secondary users. We essentially show that our approximation algorithms satisfy a monotone property.

The rest of the paper is organized as follows. In Section 2, we define the problems to be studied in this paper. From Section 3 to Section 6, we discuss algorithms for several versions of problems described in Section 2. Then we review related results on those spectrum assignment problems in Section 8 and conclude the paper in Section 9 with discussion of some possible future works.

2 Preliminaries

2.1 Network Model

Consider a wireless network system formed by some primary users who hold the right of some spectrum channels, secondary users $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ who want to lease the right to use some channels in some region for some time period. For simplicity, we treat all primary users as one unified central authority. In certain applications, each secondary user v_i may provide service to some clients within a geometry region. Let $\mathcal{F} = \{\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_m\}$ be the set of m frequencies that can be used by some secondary users for a given time interval $[0, T]$. For some wireless network systems, it is possible that the primary users will only lease a spectrum frequency for a certain time interval in a certain geographical region. If this is the case, we assume that for each $\mathbf{f}_i \in \mathcal{F}$, we associate it with a region Ω_i and a set of time intervals $\mathcal{T}_i$ that it is available. In this paper, most of our results assume that every channel will be available everywhere and everytime. Our results can easily deal with a general Ω_i and $\mathcal{T}_i$.

We assume that a secondary user v_i may wish to lease a set of channels $\mathcal{F}_i \in \mathcal{F}$. For a bidding, the secondary user will also specify two additional constraints: space condition and time condition. Each user v_i will specify a 2D region exclusively which is typically a disk $D(v_i, r_i)$ centered at node v_i with a radius r_i. User v_i also specifies a time interval $[s_i, e_i]$ or a time duration d_i exclusively. Here it is assumed that $0 \leq s_i < e_i \leq T$ and $0 < d_i \leq T$. Generally, we use T_i to denote the time constraint of user v_i, where T_i is either $[s_i, e_i]$ or a scalar $d_i > 0$.

Two different models of secondary users will be studied in this paper. The first model assumes that every secondary user is *single-minded*: when user v_i bids for $\mathcal{F}_i$, the valuation of v_i over an assignment is 0 if not *all* frequencies in $\mathcal{F}_i$ are allocated. The secondary user v_i will be called *flexible* if it will pay the allocated frequencies separately. For a flexible user v_i, we assume that for each channel $\mathbf{f}_j \in \mathcal{F}_i$, user v_i will bid $b_{i,j}$ for the usage of channel $\mathbf{f}_j$ for a certain time and within certain region. In this case, we use $b_i = \{b_{i,1}, b_{i,2}, \cdots, b_{i,m}\}$ to denote the bid vector of user i, where $b_{i,j} = 0$ if v_i did not bid for $\mathbf{f}_j$. Thus, a bidding by a user v_i will be written as follows $B_i = [b_i, \mathcal{F}_i, D(v_i, r_i), T_i]$. Upon receiving the bids from secondary users, the central authority decides an allocation method $X = \{x_1, x_2, \cdots, x_n\}$ where $x_i \in \{0, 1\}$ denotes whether user v_i's bid will be satisfied, and also a time-interval $[\mathbf{s}_i, \mathbf{e}_i]$ with $\mathbf{e}_i - \mathbf{s}_i = d_i$ when user v_i required a time-duration d_i in the bid B_i. The allocation must be conflict free among satisfied bids. Here two bids B_i and B_j conflict if $\mathcal{F}_i \cap \mathcal{F}_j \neq \emptyset$, $D(v_i, r_i) \cap D(v_j, r_j) \neq \emptyset$, and $[\mathbf{s}_i, \mathbf{e}_i] \cap [\mathbf{s}_j, \mathbf{e}_j] \neq \emptyset$. The objective of an allocation is to maximize $\sum_{i=1}^{m} x_i b_i$. For simplicity, given a set of bids Y, we use $w(Y)$ to denote the total weight of bids in Y, *i.e.*, $\sum_{B_i \in Y} b_i$.

2.2 Problems Formulation

In this paper, we study several versions of spectrum assignment problems by separately assuming channel, region and time requirements. For notational convenience, we use CRT to denote a problem, where

- C denotes channel requirements. Here C will be either S (denoting that secondary users are single-minded), or F (denoting that secondary users are flexible), or Y (denoting that there is only one channel available).
- R denotes region requirement. Here R will be either O (denoting that required regions overlap) or U (denoting that required regions are unit disks), or G (denoting that required regions are disks with arbitrary radii).
- T denotes time requirement. Here T will be either I (denoting that each required time is an interval) or D (denoting that each required time is a duration) or M (denoting that each required time is an interval or a duration).

For example, problem SUI represents the case that each user v_i will bid for a subset of channels $\mathcal{F}_i$ and is single-minded, will require a unit disk region $D(v_i, 1)$, and a fixed time-interval $[s_i, e_i]$. For each problem where each secondary user bids separately for each channel (C = F), we have a corresponding C = Y problem when considering each user requires k channels as k separate users each requires a different channel. Therefore, we don't discuss these C = F problems as they are special cases of C = Y problems.

Some versions of the problems turn out to be some well-studied problems in the literature and some well-studied problems turns out to be a special case of the above problems. Problem YOD is essentially a knapsack problem, which has a well-known FPTAS [5]. Maximum weighted independent set of a disk graph is a special case of problem YGI with $e_i - s_i \geq T/2$ for each secondary user i. The multi-knapsack problem is also a special case of problem YUD. Due to wireless network applications, we will mainly focus on the problems YOM, YUI, YUD, YUM, SUI.

3 Algorithm for Problem YOM

In this section, we design an approximation algorithm for problem YOM, where there is only one channel available, required regions overlap, and the required time is an interval or a duration. By using FPTAS of knapsack problem and dynamic programming, we can get a simple $(1-\epsilon)/2$ approximation algorithm as follows. We partition the users into two groups: one group of users who required some fixed time intervals and the other group of users who required some time intervals. We solve the assignment problem for each group and take the better assignment as the final solution.

4 PTAS for Problem YUI

In this section, we present a PTAS for the problem YUI, where there is only a single channel available, the required region is a unit disk, and the required time is an interval. The PTAS runs in $O(n^{\frac{1}{\epsilon^2}})$ time and provides approximation factor of $(1-\epsilon)$ where n is the number of secondary users.

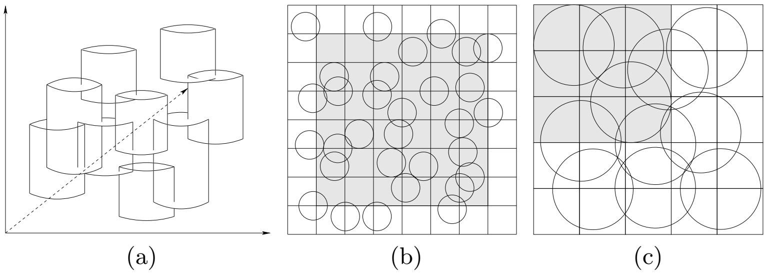

Fig. 1. (a) An illustration of cylinder graph; (b) the network at the end of ith iteration; (c)An illustration of space using hyperplanes. This is a view from the bottom (Z-axis).

Notice that finding the set of bids with the maximum value is equivalent to solve the maximum weighted independent set in the following intersection graph of cylinders. Each bid $B_i = [b_i, \mathcal{F}_i, D(v_i, r_i), T_i]$ defines a cylinder $\mathbf{B}_i = (D(v_i, r_i) \times [s_i, e_i])$ with weight b_i. See Figure 1(a) for illustration. For simplicity, we assume that the three axes are X, Y and Z and the axis Z denotes the time dimension. The disk $D(v_i, r_i)$ is called the base of the cylinder $\mathbf{B}_i$.

Our PTAS is based on the shifting strategy developed in [9]. We partition the space using hyperplanes perpendicular to X-axis and hyperplanes perpendicular to Y-axis. See Figure 1(b) for illustration. In each partition, we throw away some cylinders that intersect some special hyperplanes and then solve the sub-instances of cylinders contained in each cell individually. Let I be the set of all n cylinders. For any give integer $k > 1$, we derive k^2 polynomially solvable sub-instances from the given instance I in polynomial time. The best value of those sub-instance solutions is at least $(1-\frac{2}{k})w(OPT(I))$, where $OPT(I)$ is the optimal solution on I and $w(OPT(I))$ is the value of the solution. Then, solve these sub-instances using a dynamic programming procedure and return the solution with best value.

4.1 Deriving sub-instances

For simplicity, we assume that diameter of each disk is 1 and each disk is open disk. Draw a grid consisting of hyperplanes $x = i$ (for $i \in Z$) perpendicular to X-axis and hyperplanes $Y = j$ (for $j \in Z$) perpendicular to Y-axis. The distance between every two parallel neighbor hyperplanes is the diameter of unit disk. So each cylinder will be hit by at most one hyperplane perpendicular to X-axis and at most one hyperplane line perpendicular to Y-axis.

For each i, j belong to $\{0, 1, \cdots, k-1\}$, we compose a sub-instance $I_{i,j}$ containing all cylinders except those being hit by a hyperplane from $\{x = p \mid p \mod k = i\}$ or hit by a hyperplane from $\{y = p \mid p \mod k = j\}$. There are k^2 different sub-instances. For each sub-instance, we calculate its optimal solutions using the dynamic programming in every of the $k \times k$ grids, which will be described in detail in the next subsection. We first establish a technical lemma for the performance guarantee (See [3] for proofs).

Lemma 1. *For at least one sub-instance $I_{i,j}$, $0 \le i, j < k$, the value of the solution $w(OPT(I_{i,j})) \ge (1-\frac{2}{k})w(OPT(I))$*

For each sub-instance, each disk is in a $k \times k$ grid. Disks in different grid won't intersect each other. So the union of solutions of all $k \times k$ grids is independent. We show that the problem in a $k \times k$ grid is polynomially solvable as follows.

4.2 Dynamic programming

We describe a dynamic programming approach to find a maximum weighted independent set for an instance $I_{i,j}$, which only considers cylinders contained in one $k \times k$ cell of $I_{i,j}$. For these cylinders, we sort them in non-decreasing order of their ending time e_i. For simplicity, let $\mathbf{B}_1$, $\mathbf{B}_2$ $\cdots$, $\mathbf{B}_n$ be the n cylinders contained in one cell in the sorted order. In the rest of subsection, we use i to denote the cylinder $\mathbf{B}_i$.

Definition 1. ***Pile**: A pile is an ordered collection $\langle j_1, j_2, \cdots, j_q \rangle$ of pairwise non-intersecting cylinders that intersect a common hyperplane $z = b$ (for some value b) perpendicular to Z-axis. Here j_i is the* ending *time of a cylinder $\mathbf{B}_{j_i}$ and $j_t < j_{t+1}$.*

Given a hyperplane $z = b$ for some fixed value b, in a pile that was hit by $z = b$, there are at most $2k^2$ cylinders in the pile. Notice that all cylinders in this pile intersect the hyperplane $z = b$ and are disjoint from each other. Then an area argument implies that the number of cylinders in a pile is at most $k^2/\frac{\pi}{4} < 2k^2$.

Lemma 2. *The total number of all possible piles in each $k \times k$ grid is polynomial of n.*

Proof. Please see [3].

Our dynamic programming approach will first sort the piles based on an order defined below; then find the optimum solution of all cylinders that are ordered in front of a pile.

Definition 2. *Define an* order *on the set of piles as following: $\langle j_1, j_2, \cdots, j_l \rangle \prec \langle i_1, j_2, \cdots, i_m \rangle$ if (1) $j_l < i_m$, or (2) $j_l = i_m$ and $\langle j_1, j_2, \cdots, j_{l-1} \rangle \prec \langle i_1, j_2, \cdots, i_{m-1} \rangle$, or (3) $m = 0$.*

Definition 3. *We define $OPT(X \mid j_1, j_2, \cdots, j_l)$ to be the maximum total weight of pairwise non-overlapping cylinders from set X given that cylinders $j_1, j_2, \cdots, j_l$ already occupy their places. For a set of non-overlapping cylinders $\{j_1, j_2, \cdots, j_l\}$ and a cylinder t with $t < j_1$, we define the marginal contribution, denoted as $R_{j_1, j_2, \cdots, j_l}(t)$, of cylinder t to $OPT(i : i \leq t \mid j_1, j_2, \cdots, j_l)$ as $(OPT(i \leq t | j_1, j_2, \cdots, j_l) - OPT(i < t | j_1, j_2, \cdots, j_l))^+$. Here $(x)^+ = \max\{0, x\}$.*

Based on the above definition, a cylinder t has positive marginal contribution $R_{j_1, j_2, \cdots, j_l}(t)$ will clearly be used in an optimum solution $OPT(i \leq t | j_1, j_2, \cdots, j_l)$. If its marginal contribution is 0, then it means that there is one optimum solution $OPT(i \leq t | j_1, j_2, \cdots, j_l)$ that will not use the cylinder t.

As proved in [11], it is easy to show that $OPT(i \leq t | j_1, j_2, \cdots, j_l) = \sum_{i=1}^{t} R_{j_1, j_2, \cdots, j_l}(i)$ and $R_{j_1, j_2, \cdots, j_l}(t) = b_t + OPT(i < t | j_1, \cdots, j_l, t) - OPT(i < t | j_1, \cdots, j_l)$.

We then present our dynamic programming to find an optimal solution in each $k \times k$ cell as following Algorithm 1.

Algorithm 1 Find Maximum Weighted Independent Set of Cylinders in Each Cell

```
Input: A set $S$ of weighted cylinders contained in a cell.
Output: An optimum maximum weighted independent set in $S$.
Find all the piles $\mathcal{P}$ of cylinders from $S$ and sort them in the order of $\prec$;
Take the piles one by one in the order starting from the least one. For each pile $\langle j_1, j_2, \cdots, j_l \rangle$ calculate and store in the memory the value $R_{j_2, \cdots, j_l}(j_1)$ according to the formula: $R_{j_2, \cdots, j_l}(j_1) = \left(b_{j_1} + \sum_{i: i < j_1} R_{j_1, j_2, \cdots, j_l}(i) - \sum_{i: i < j_1} R_{j_2, \cdots, j_l}(i)\right)^+$
After all the piles corresponding values of $R_{j_2, j_3, \cdots, j_l}(j_1)$ is calculated for every possible pile $p = \langle j_1, j_2, \cdots, j_l \rangle \in \mathcal{P}$, we schedule the cylinders in the following way. Consider cylinders from $S$ in the order of decreasing number. Take a cylinder $j$ next in that order. Suppose that cylinders $\{j_1, j_2, \cdots, j_l\}$ are already scheduled, then schedule $j$ iff $R_{j_1, j_2, \cdots, j_l}(j)$ is positive.
```

Theorem 1. *The running time of our dynamic programming is at most $O(n^{2k^2+1})$.*

Proof. Please see [3].

Then by setting $k = 2/\epsilon$, our method implies a PTAS in time $n^{\frac{1}{\epsilon^2}}$.

5 Algorithm for Problem YUD and YUM

In this section, we design approximation algorithms with a constant approximation ratio for problems YUD and YUM respectively.

For YUD, we assume there is only one channel available, the required region is a unit disk and required time is a duration. We can use the same graph of cylinders mentioned above for illustration. The main idea here is to partition cylinders into g groups. Solve the maximum weighted independent set in each group and then take the group with the best solution. By pigeonhole principle, it will give us $1/g$ approximation. We mainly will focus on designing a partition with minimum constant value g, please see [3] for details.

For YUM, where there is only one channel available, the required region is a unit disk and required time is a duration or an interval. The main idea is to partition users into 2 groups. One group G_d includes all users asking for a time duration and the other group G_i includes all users asking for a time-interval. We solve group G_d by the algorithm for YUD, solve group G_i by the PTAS for YUI, and then take the group with the best solution. It is a $1/10$ approximation algorithm for problem YUM, please see [3] for details.

6 Algorithm for Problem SUI

In this section, we design a $\Theta(\sqrt{m})$-approximation algorithm for problem SUI. Notice that the set packing problem is a special case of the problem SUI due to the observation in [3]. Recall that set packing problem is not approximable within $m^{1/2-\varepsilon}$ for any $\varepsilon > 0$, unless NP=ZPP. Thus, we have

Theorem 2. *Problem* SUI *is not approximable within $m^{1/2-\varepsilon}$ for any $\varepsilon > 0$, unless NP=ZPP.*

Assume that each bidder bids at most k frequencies and single time interval.

When $k = 1$, obviously the optimum solution is the union of the optimum solutions OPT_i, where OPT_i is the optimum solution for the set of users who do bid for frequency $\mathbf{f}_i$. We also use OPT to denote the global optimal solution hereafter. Notice that OPT_i can be solved by dynamic programming.

When $k > 1$, for each frequency $\mathbf{f}_i$, let $OPT|_i$ be the set of users in OPT that bid for frequency $\mathbf{f}_i$. Obviously, we have $\sum_{i=1}^{m} OPT|_i \geq k \times OPT$, since each user in OPT will appear in at least k different $OPT|_i$. Thus, $\max\{OPT|_1, OPT|_2, \cdots, OPT|_m\} \geq \frac{k}{m} \times OPT$.

We partition the bidders into two groups:

1. Z_1 contains all the bidders that bid at least $\sqrt{m}$ frequencies;
2. Z_2 contains all the other bidders, *i.e.*, bid less than $\sqrt{m}$ frequencies.

We approximate optimal solution for each group and return the larger one. We will show $\Theta(\sqrt{m})$-approximate algorithms for both groups respectively. Then, the maximum of these two solution must give us a $\Theta(\sqrt{m})$ approximation.

First for group Z_1, we just use $\max\{OPT_1, OPT_2, \cdots, OPT_m\}$ as our solution. Since each bidders bids at least $k \geq \sqrt{m}$ frequencies, $\max_{i=1}^{m} OPT(Z_1)|_i \geq$

$\frac{\sqrt{m}}{m}OPT(Z_1) = \frac{1}{\sqrt{m}}OPT(Z_1)$. Here $OPT(Z_j)$ is the optimal solution for users in group Z_j for $j = 1, 2$. For bidders in Z_1, we use I_i to denote the set of bidders in Z_1 that bid for frequency $\mathbf{f}_i$. Then we find the maximum weighted independent set OPT_i of I_i using standard dynamic programming. Obviously, $OPT_i \geq OPT(Z_1)|_i$. Thus, $\max_{i=1}^{m} OPT_i \geq \max_{i=1}^{m} OPT(Z_1)|_i \geq \frac{1}{\sqrt{m}}OPT(Z_1)$.

For group Z_1, we convert this problem into Scheduling Split Intervals Problem (SSIP) [1]. The ordinary SSIP considers scheduling jobs that are given as groups of non-intersecting segments on the real line. Each job J_i is associated with an interval, I_j, which consists of up to t segments and a positive weight, w_j. Two jobs are in conflict if any of their segments intersect. The objective is to schedule a subset of non-conflicting jobs with maximum total weight.

A $2t$-approximation algorithm for problem SSIP is given in [1]. Here we create a special instance for SSIP problem as follows. Let $[0, T]$ be the original time period where bidders can place their time-interval. We then create a bigger time period $[0, m \cdot T]$ where m is the total number of frequencies. Then a user i will be associated with following $t_i = |\mathcal{F}_i| < \sqrt{m}$ segments in $[0, m \cdot T]$: $\{[(j-1) \cdot T + s_i, (j-1) \cdot T + e_i] \mid \mathbf{f}_j \in \mathcal{F}_i, 1 \leq j \leq m\}$ In other words, we duplicate the period $[0, T]$ m times for each of the frequencies in $\mathcal{F}$, and a user i will have a segment in the jth duplication if it bids for frequency $\mathbf{f}_j$. Then there are at most $\sqrt{m}$ segments for every bidder. Then based on algorithms in [1], we get $2\sqrt{m}$-approximation solution, *i.e.*, find a solution with value at least $\frac{OPT(Z_2)}{2\sqrt{m}}$.

Then the maximum of the above two solutions is at least $\max\left(\frac{OPT(Z_1)}{\sqrt{m}}, \frac{OPT(Z_2)}{2\sqrt{m}}\right) \geq \frac{1}{3\sqrt{m}}OPT$, since either $OPT(Z_1) \geq \frac{1}{3}OPT$, or $OPT(Z_2) \geq \frac{2}{3}OPT$ from $OPT(Z_1) + OPT(Z_2) \geq OPT$.

Notice that by using a different group partition, where group Z_1 contains the bidders that bid for at least $\sqrt{\frac{m}{2}}$ frequencies, we get an algorithm with approximation ration $\frac{\sqrt{2}}{4\sqrt{m}}$.

As a byproduct of our above results, we show that for problem SSIP, there is no approximation algorithm with ratio $O(t^{1-\epsilon})$ for any $\epsilon > 0$ unless $NP = ZPP$.

Theorem 3. *For problem SSIP, there is no polynomial-time approximation algorithm with ratio $O(t^{1-\epsilon})$ for any $\epsilon > 0$ unless $NP = ZPP$.*

Proof. Please see [3].

We can extend the above case. such that each bidder requires at most t time intervals. Obviously, $t \leq \lfloor |\mathcal{F}|/2 \rfloor$, every pair of users have a common requested frequency. Thus, for this case, computing the approximation solution for each frequency is exactly the traditional SSIP. We can get $2t$-approximation solution for this special case. Otherwise, we have a $\Theta(t\sqrt{m})$-approximation algorithm by using similar method above, please see [3] for details.

7 Strategyproof Mechanism Design

In this section, we show how to design a straightforward mechanism, *i.e.*, for each secondary user, reporting its valuation truthfully maximizes its profit, based on the algorithms discussed in previous sections. A strategyproof mechanism can designed by using monotone output algorithm and critical value payment scheme. An output algorithm is monotone if an agent will still participate in the output when it increase its bid. We define a critical value θ_i, *i.e.*, the minimum valuation v_i that makes agent i participate in the output. A critical value payment scheme P^O for an algorithm O such that $p_i = \theta_i$ if agent i is in the output, otherwise, $p_i = 0$. If O is a monotone algorithm and P^O is a critical value payment scheme for O, $M = (O, P^O)$ is a strategyproof mechanism.

In the algorithms mentioned above, we used technology of dynamic programming, grouping and the classic FPTAS for knapsack problem. Dynamic programming and grouping do not affect the monotone property. However, the classic FPTAS for knapsack problem is not monotone. A counterexample is given in [3]. In [4], Briest proposed an alternative rounding scheme that transform a pseudopolynomial algorithm into a monotone FPTAS for knapsack problem. Using this FPTAS for knapsack problem, all algorithms in this paper are monotone. Therefore we can design strategyproof mechanisms $M = (O, P^O)$ for all problems discussed in previous sections.

8 Literature Reviews

The problems we discussed above are at the intersection of a lot of famous problems. Here we review results for some of these problems.

Knapsack problem, which is same as simple problem YOD, has a classic FPTAS by the means of rounding. However, we cannot design a strategyproof mechanism using this FPTAS since it is not monotone. An alternative rounding scheme was proposed by Briest in [4], which gave a new rounding scheme leading to a monotone FPTAS for knapsack problem.

In [10], Jansen and Zhang presented a $(2 + \epsilon)$-approximation algorithm for rectangle packing problem which is similar with our problem YUI. If we don't consider intersections in space, it is a unit-height rectangle packing problem, which is a special case of rectangle packing. Kovaleva described a PTAS for unit-height rectangle packing problem in [11]. We extend this PTAS to a PTAS for problem YUI as described above.

Problem SUI is a special case of set packing problem. In [8], Hastad proved that set packing problem cannot be approximable within $m^{\frac{1}{2}-\epsilon}$, unless $NP = ZPP$. Another special case of set packing problem is Scheduling Split Intervals Problem(SSIP). Bar-Yehuda *et al.* [1] gave a $2t$-approximation algorithm. We show how to convert the problem SUI to SSIP in previous section.

9 Conclusions

In this paper, we combine the game theory with communication modeling to solve the channel assignment problem. We study how to assign the spectrum and charge the secondary users such that the *overall social benefits* is maximized.

More specifically, we formalize several versions of spectrum assignment problems by separately assuming channel, region and time requirements. We also show how to design strategyproof mechanism based those algorithms. We leave it as a future work whether there are efficient approximation algorithms for problem SUD, where single-minded secondary users request unit disk region and a time duration, whether there are PTASs for problems YOM, YUD and YUM.

References

1. Bar-Yehuda, R., Halldórsson, M. M., Naor, J. S., Shachnai, H., and Shapira, I. Scheduling split intervals. In *ACM SODA* (2002), pp. 732–741.
2. Bartal, Y., Gonen, R., and Nisan, N. Incentive compatible multi unit combinatorial auctions. In *TARK '03: Proc. of the 9th conf. on Theoretical aspects of rationality and knowledge* (2003), pp. 72–87.
3. X.Y, Li., P, Xu., S.J, Tang.and X.W, Chu. Spectrum Bidding in Wireless Networks and Related. *Technical Report*, IIT, March 30 2008 .
4. Briest, P., Krysta, P., and Vöcking, B. Approximation techniques for utilitarian mechanism design. In *ACM STOC* (2005), pp. 39–48.
5. Chekuri, C., and Khanna, S. A PTAS for the multiple knapsack problem. In *ACM SODA* (2000), pp. 213–222.
6. Clarke, E. H. Multipart pricing of public goods. *Public Choice* (1971), 17–33.
7. Groves, T. Incentives in teams. *Econometrica* (1973), 617–631.
8. Hastad, J. Clique is hard to approximate within $n^{1-\varepsilon}$. *Acta Mathematica* (1999), 105–142.
9. Hochbaum, D. S., and Maass, W. Approximation schemes for covering and packing problems in image processing and VLS. *Journal of ACM 32* (1985), 130–136.
10. Jansen, K., and Zhang, G. On rectangle packing: maximizing benefits. In *ACM SODA* (2004), pp. 204–213.
11. Kovaleva, S. Improved dynamic programming subroutine in the PTAS for the unit-height rectangle packing problem. In *APPOL II Workshop* (2002).
12. Lehmann, D. J., O'Callaghan, L. I., and Shoham, Y. Truth revelation in approximately efficient combinatorial auctions. In *ACM Conf. on Electronic Comm.* (1999), pp. 96–102.
13. Osborne, M. J., and Rubinstein, A. *A course in game theory.* The MIT Press, 2002.
14. Papadimitriou, A. A. C., Talwar, K., and Tardos, E. An approximate truthful mechanism for combinatorial auctions with single parameter agents. in *ACM SODA* (2003), pp. 205–214.
15. Stine, J. A. Spectrum management: The killer application of ad hoc and mesh networking. In *IEEE Symp. on New Frontiers in Dynamic Spectrum Access Net.* (2005).
16. Vickrey, W. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance* (1961), 8–37.