Two-Dimensional Packing Problems in Telecommunications

Silvano Martello
DEIS, University of Bologna, Italy
silvano.martello@unibo.it

from joint works with A. Lodi (University of Bologna) & M. Monaci (University of Padova)

together with

C. Eklund & Jani Moilanen (Nokia Siemens Networks)

C. Cicconetti, L. Lenzini & E. Mingozzi (Univ. Pisa)

and

C. Hurkens & G. Woeginger (TU Eindhoven)

April 2012, Izmir

Outline of this talk

- **Objective:** description of the development of an interdisciplinary research applicable to real world problems.

- **Four teams involved:** in chronological order,

  Nokia Siemens laboratory: research group on the *IEEE 802.16/WiMAX standard*;

  University of Pisa: research group on *Computer Networking* (**Prof. Luciano Lenzini**);

  University of Bologna: research group on *Combinatorial Optimization* (**S.M.**);

  Technical University of Eindhoven: research group on *Theoretical Combinatorial Optimization* (**Prof. Gerhard J. Woeginger**).

- The whole project has been described in:

  Lodi, Martello, etc ... Efficient two-dimensional packing algorithms for mobile WiMAX. *Management Science*, 2011.

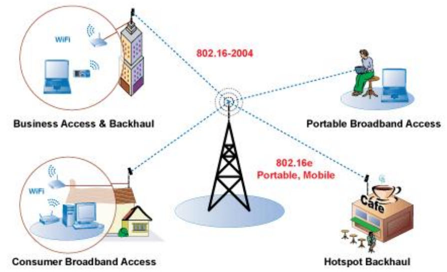
Contents

The project has been developed following the classical steps of an applied research:

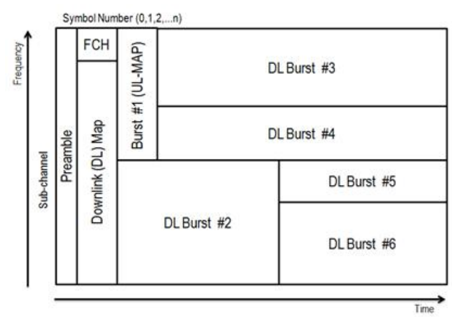
1. **birth:** a real-world problem;
2. development of mathematical **models** (new two-dimensional packing problems);
3. theoretical analysis (**computational complexity**: $\mathcal{NP}$-hard problems);
4. definition of mathematical models for **the real-world problems**;
5. evaluation of the **technological constraints** (extremely tough CPU limitations);
6. development of solution **algorithms** (fast and efficient heuristics);
7. implementation and **experimental evaluation** on realistic scenarios.

1. The birth: an optimization problem in telecommunications

Telecommunication systems adopting the **IEEE 802.16/WiMAX** standard:



- a fixed station transmits/receives data packets to/from other stations (e.g., the mobile phones);
- all transmissions are performed using rectangular frames [time × frequency] (**downlink zones**).

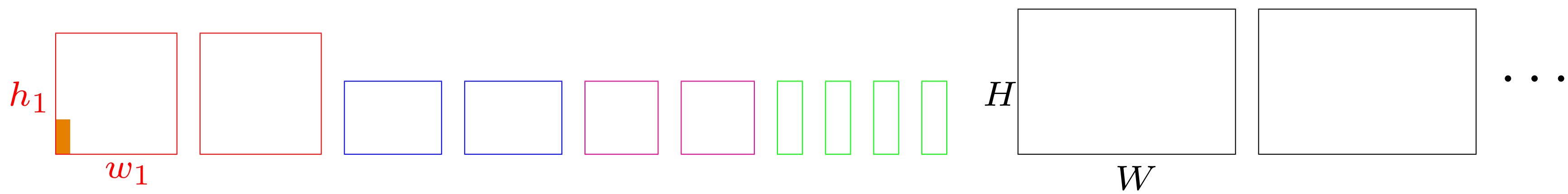


- The fixed station must maximize the frame utilization by
  1. deciding which packets will be included in the next transmission phase;
  2. arranging each selected packet into one or more rectangular regions;
  3. allocate the regions to the frame (without overlapping).

2. The models: a look at the combinatorial optimization literature

Classical **Two-Dimensional Bin Packing Problem (2BP)**

- given n **rectangles** (**items**), having width w_j and height h_j $(j = 1, \ldots, n)$,



- and an unlimited number of large rectangles (**bins**), having width W and height H,
- **A. pack all the items**, without overlapping, in the **minimum number of bins**:

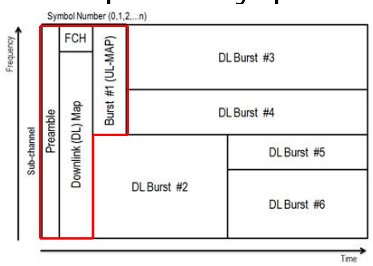
- **B. pack a subset of items**, without overl., in a **single bin maximizing the packed area**.
- **Many variants**: The items may/may not be **rotated**; by **90°/any angle**;

  guillotine cutting may/may not be imposed (items must be obtained through a sequence of edge-to-edge cuts parallel to the edges of the bin);
- . . . **large literature**
- Generalization of the **One-Dimensional BP**: n items of size w_j, bins of size W.

2. The models: our problems vs standard 2BPs

Main difference

- **Input to 2BP**: set of rectangles to be packed.
- **Input to the telecommunication problems**: set of data packets to be packed:
  - a **data packet** is an amount of information, in practice a **number**;
  - this number may be interpreted as an **area** a_j;
  - this area must be allocated to a $w_j \times h_j$ rectangle such that $w_j h_j \geq a_j$,
  - **or** to a number m_j of rectangles such that $w_{j_1} h_{j_1} + \ldots + w_{jm_j} h_{jm_j} \geq a_j$;
  - the selected rectangles must then be optimally packed in the **downlink zone** (the **bin**):



  - each packed rectangle needs information in the downlink zone (sizes, coordinates), i.e.,
  - part of the bin is used for **maps transmission**: size proportional to number of rectangles;
  - hence the need of limiting the number of rectangles.

3. Theoretical analysis: Problem P0

Questions:

- How difficult are the telecommunication problems at hand?
- Can they be solved in polynomial time? If not
- Can they be solved in pseudo-polynomial time? If not
- Can they be approximated with worst-case performance guarantee in polynomial time?
- Can they be solved efficiently in practice?

To answer these questions, let us consider the simplest combinatorial optimization problem we can "extract" from the given problems:

Problem P0 (Area Packing):

- n areas;
- a single bin;
- allocate each area to one rectangle, and

  pack all the rectangles into the bin without overlapping.

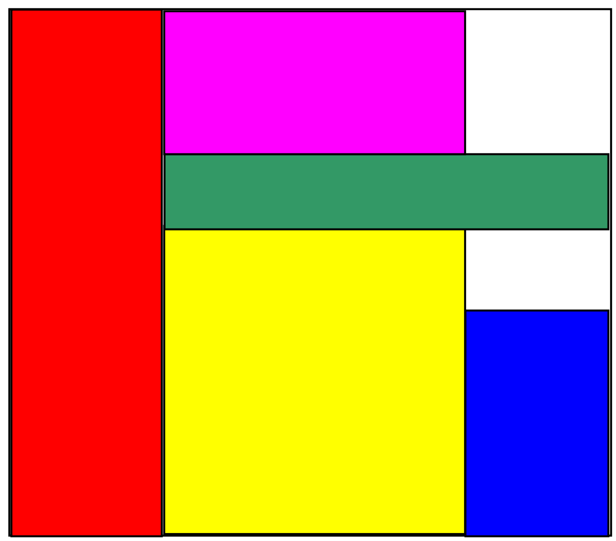
3. Theoretical analysis: recognition version of P0

- Formally:
  - n integer areas $a_j, \ j \in J = \{1, \ldots, n\}$ and
  - a single bin of integer sizes $W \times H$, with $W \cdot H \geq \sum_{j \in J} a_j$
- Is it possible to find integers $w_1, \ldots, w_n$ and $h_1, \ldots, h_n$ such that:
  - $a_j = w_j h_j, \ j \in J$, and
  - the n rectangles $R_j = [w_j, h_j], \ j \in J$, can be packed into the bin without overlapping?

- 8 x 7

- 14
- 6
- 8
- 6
- 16

- 14 = 2 x 7
- 6 = 2 x 3
- 8 = 4 x 2
- 6 = 6 x 1
- 16 = 4 x 4



3. Theoretical analysis: Complexity of P0

- A simple (although non-trivial) transformation from a variant of PARTITION shows that **P0 is ordinary NP-complete**.
- Sophisticated techniques using
  - tools from number theory,
  - transformation from a variant of THREE-PARTITION

  prove that **P0 is strongly NP-complete**.

  Hurkens, Lodi, Martello, Monaci and Woeginger
  Complexity and approximation of an area packing problem
  Optimization Letters, 2012.
- Hence P0 **cannot be solved** in polynomial time, nor in pseudo-polynomial time unless $\mathcal{P} = \mathcal{NP}$.
- However its **optimization version** can be approximated with worst-case performance guarantee in polynomial time.

3. Theoretical analysis: Optimization version of P0

- The recognition version of problem P0 can be transformed into the following **optimization version**:
  - assume that any area a_j can be arbitrarily **split** into integer rectangular sub-areas (at most a_j 1×1 (unit) squares);
  - in this way the problem always has a feasible solution;
  - objective: pack all areas into the bin without overlapping by **minimizing the number of created rectangular sub-areas**.

- Of course, if the optimal solution to the optimization version has value n, i.e., a unique rectangular sub-area is created for each original area, then the recognition version has answer "YES".

- This version makes sense by itself as a very naïve approximation of the application at hand. In other words, the best configuration is obtained by minimizing the number of sub-areas.

3. Theoretical analysis: a 3-approx algorithm for P0

- The general philosophy of the algorithm consists of the following phases:

A. Split each **area** a_j, $j \in J$, into **two parts**:

A.1 a **"large" rectangle** of size $\widetilde{w}_j \times H$ (H the height of the bin) , with

$$\widetilde{w}_j = \left\lfloor \frac{a_j}{H} \right\rfloor, \text{ and}$$

A.2 a one dimensional (vertical) **strip**, i.e., a rectangle of size $1 \times \widetilde{h}_j$ with

$$\widetilde{h}_j = a_j - \widetilde{w}_j H$$

(possibly only one part is created)

B. Subdivide the **bin** into **two parts** having height H:

B.1 a **"large" portion** of size $W_\ell (= \sum_{j \in J} \widetilde{w}_j) \times H$ that allocates the rectangles;

B.2 a **"small" portion** of size $W_s (= W - W_\ell) \times H$,

whose W_s columns, of size $1 \times H$, are treated as W_s 1-dimensional bins:

consecutively allocate the one dimensional strips to the 1-dimensional bins

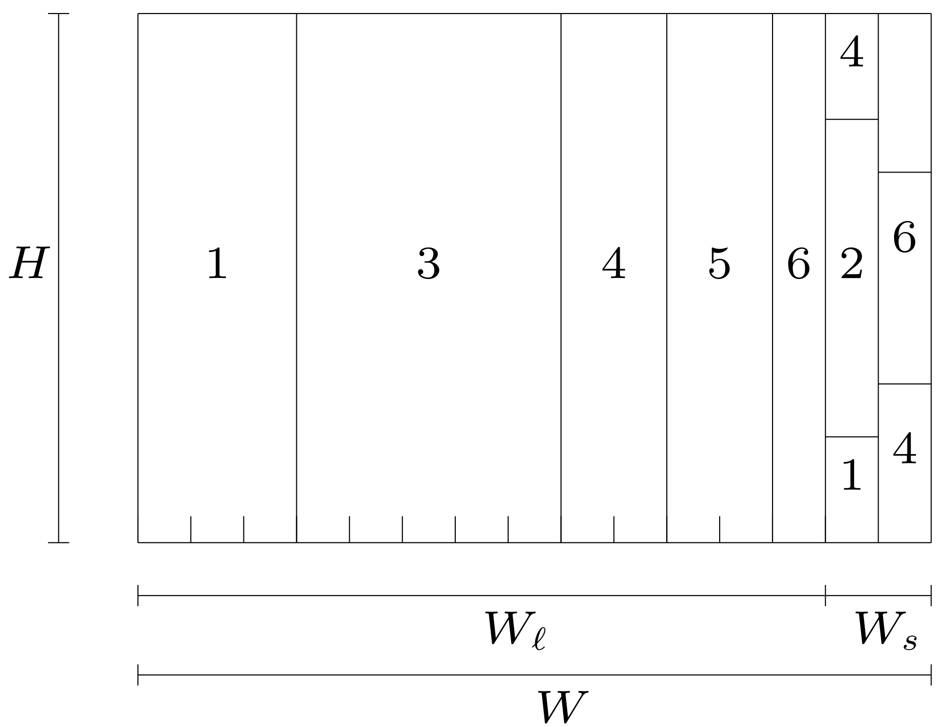
by further splitting only when necessary.

C. Post-optimize the solution. (Not needed for the worst-case guarantee.)

3. Theoretical analysis: a 3-approx algorithm for P0, example

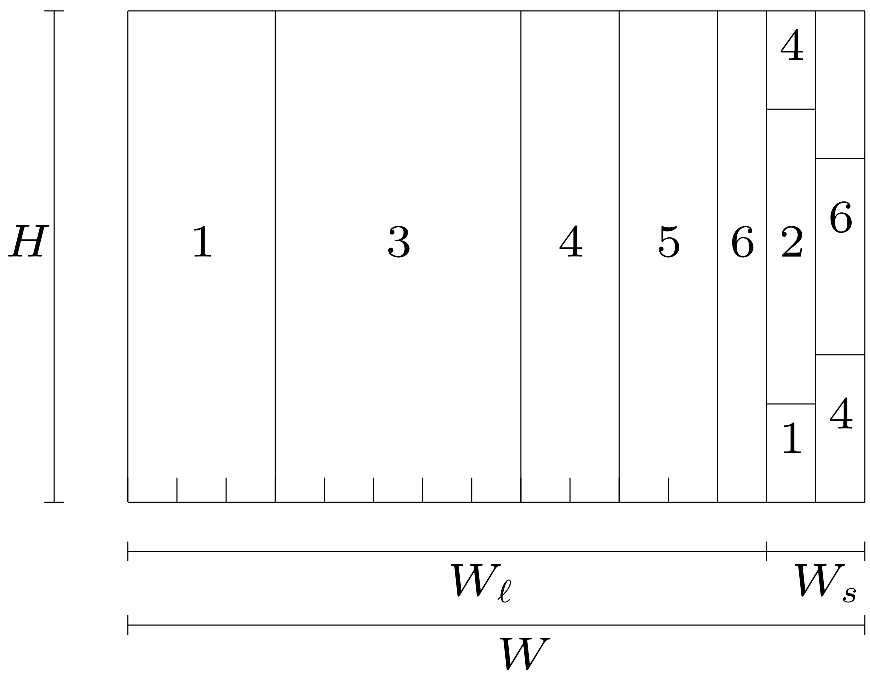
Instance with $W = 15$, $H = 10$

area	a_j	$\widetilde{w}_j$	$\widetilde{h}_j$
1	32	3	2
2	6	-	6
3	50	5	-
4	25	2	5
5	20	2	-
6	14	1	4



3. Theoretical analysis: a 3-approx algorithm for P0, proof

- Consecutively pack the strips in the first column until a strip is found that does not fit;
  split such strip, packing the largest feasible part in the current column;
  initialize the next column with the remaining part, and continue until all strips are packed.
- **Hence** each strip is split at most once (recall that each strip has size $\widetilde{h}_j < H$).
- **Hence** each area produces at most three sub-areas, which proves the worst case behavior.



- It can be shown that the bound is tight.

3. Conclusions of the theoretical analysis

Given an instance of Area Packing Problem (P0),

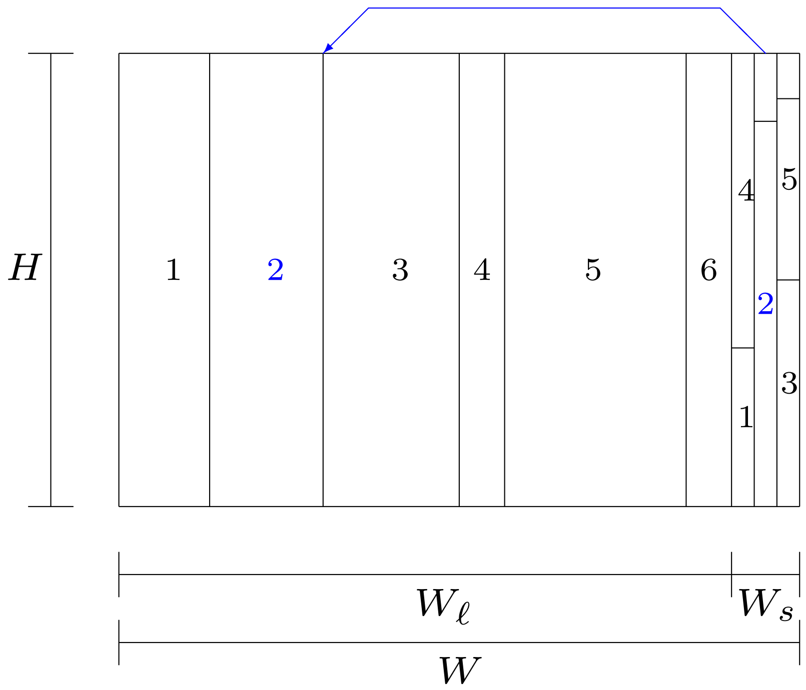
- it is **strongly NP-complete** to decide whether there is a feasible solution that has a **single rectangle per area**;
- it is trivial to construct instances for which such a solution does not exist;
- it is always possible to construct a solution that has at most **three rectangles per area**;
- such a solution can be found in **linear time**;
- what about the intermediate case (**two rectangles per area**)?
- it can be proved that all instances with $n \leq 3$ **areas** have a feasible solution with two rectangles per area;
- **Conjecture:** Every instance possesses a feasible solution with at most two rectangles per area.

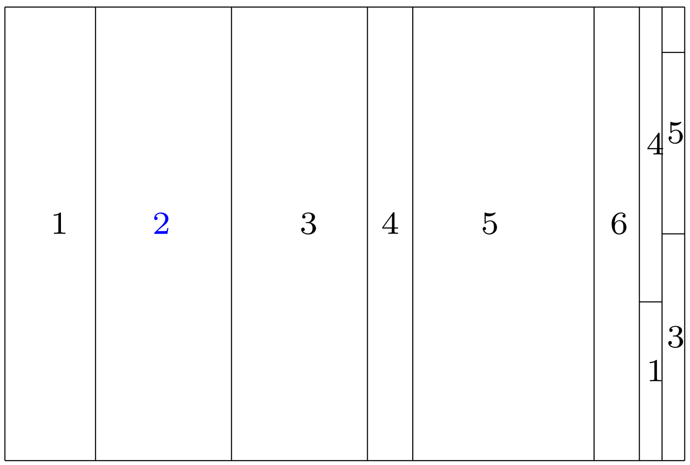
Post-optimization of the approximation algorithm

- Post-optimization is useful in practice when there are areas j such that:

  (i) both the associated large rectangle and one dimensional strip have been created, and

  (ii) strip j is packed alone in a 1-dimensional bin (column):





- Move the 1-dimensional bin that packs strip j close to the rectangle associated with area j.
- $\Rightarrow$ New solution in which area j is packed with a unique rectangle $(\widetilde{w}_j + 1) \times H$.

4. The real-world problems

Three main differences in the telecommunication problems at hand:

(I) The areas cannot be arbitrarily split:

- For each area a_j $(j \in J)$, m_j sub-areas, each having a specified integer value

$$a_{jl} \ (j \in J, l \in L_j = \{1, \ldots, m_j\}),$$

are given in input, such that

$$\sum_{l \in L_j} a_{jl} = a_j \ \forall \ j \in J$$

- The sub-areas cannot be split.
- For each area we must define one or more rectangles containing sub-areas.
- This can make it impossible to completely pack all areas.

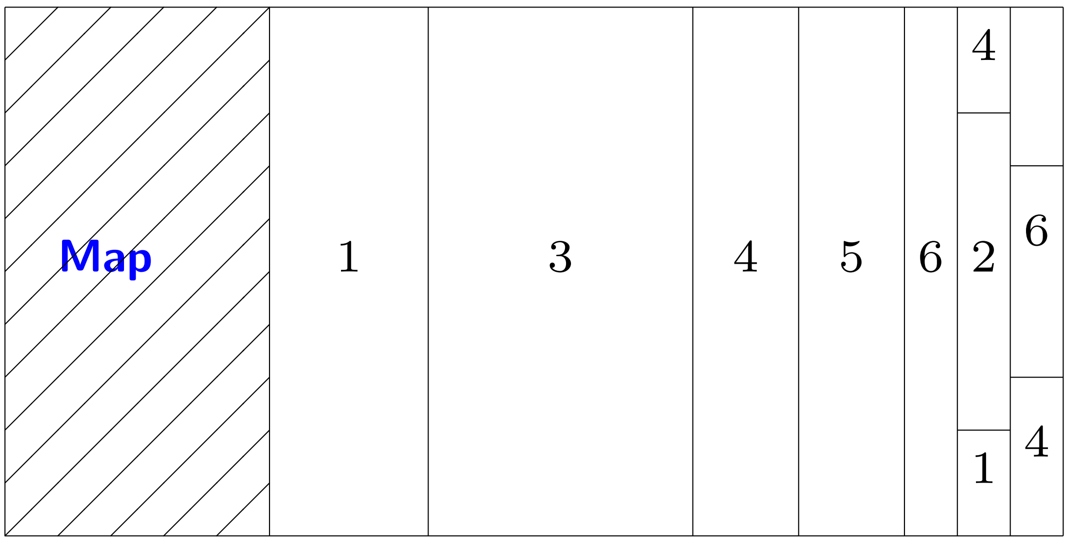
(II) Each sub-area has a profit (priority):

- The objective function is to maximize the total packed profit.

4. The real world problems

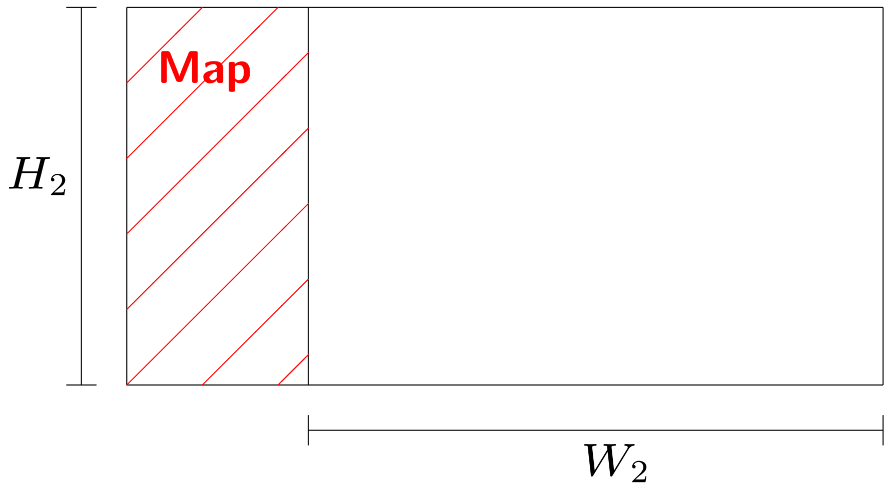
(III) The mapping of the packing must be stored in the frame.

- Each packed rectangle requires additional information (size and position of the rectangle, pointer to the associated area, . . .);



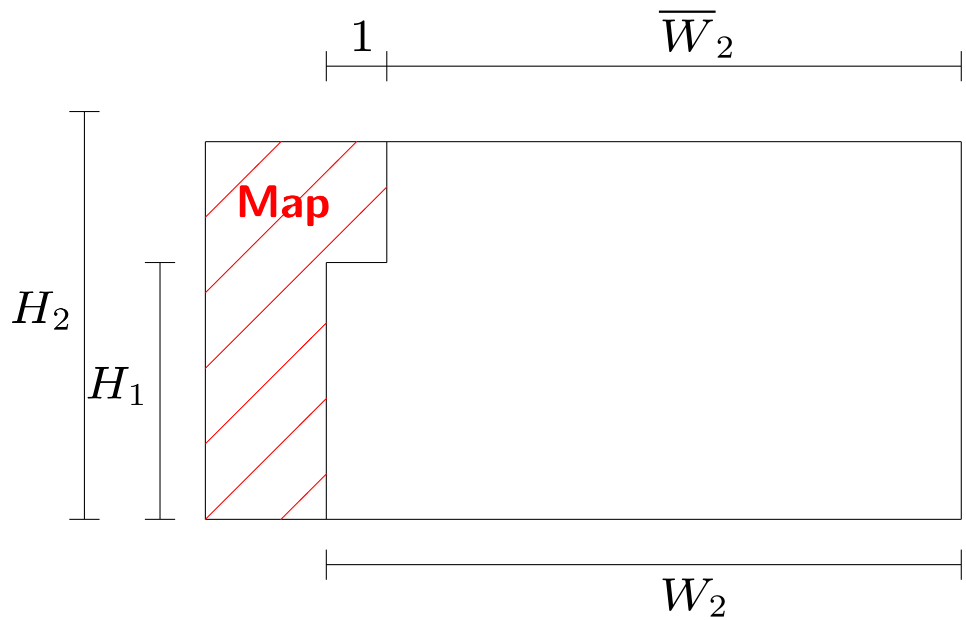
- minimizing the number of rectangles leads to minimizing the size of the map. **However.**
- **the actual size of the map can only be computed once the packing is known.**

4. Real world problems: P1 and P2 (Distributed Permutation Zone)



⟵ Bin structure for **Problem P1**

Bin structure for **Problem P2** ⟶



P2 is a generalization of **P1**

- A third real-world problem (**P3**) will be discussed later.

5. Evaluation of the technological constraints

- The planned system must use sets of standard PCs;
- each PC must perform **500 transmissions per second**, i.e.,
- every **2 milliseconds** it is necessary to
  - read the input;
  - execute the algorithm;
  - produce the output (packing and map);
  - transmit the corresponding packets.
- The bad news is that each transmission takes **1 millisecond**, i.e.,
- each instance must be completely solved (packing and map) within **1 millisecond!**.

  (Although real instances are "small", this requirement was really tough!)

C. Cicconetti, L. Lenzini, A. Lodi, S. Martello, E. Mingozzi, M. Monaci.

Efficient two-dimensional data allocation in IEEE 802.16 OFDMA

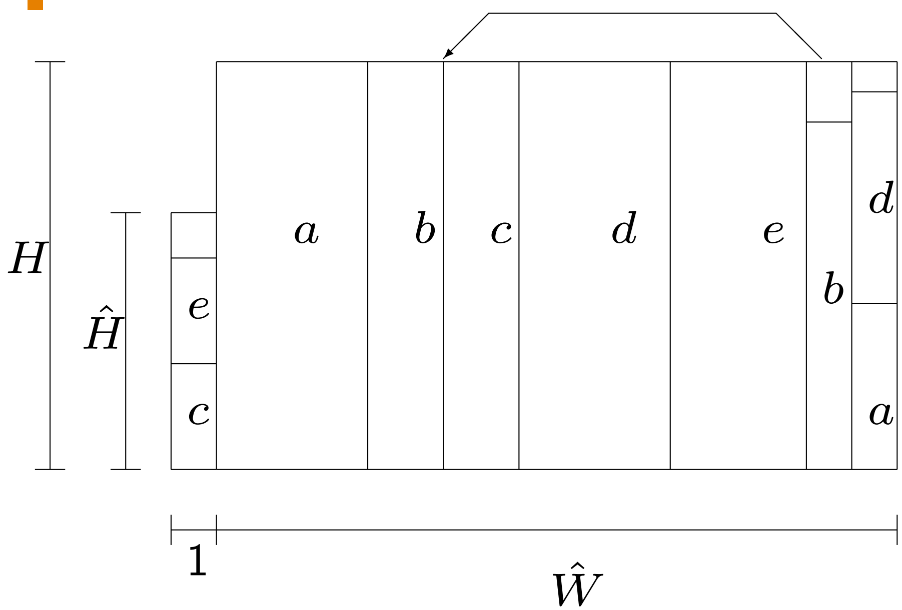
Proceedings of IEEE INFOCOM 2010.

A Fast and Efficient Algorithm to Exploit Multi-user Diversity in IEEE 802.16 BandAMC.

Computer Networks, 2011.

6. Development of heuristic algorithms: Stripes

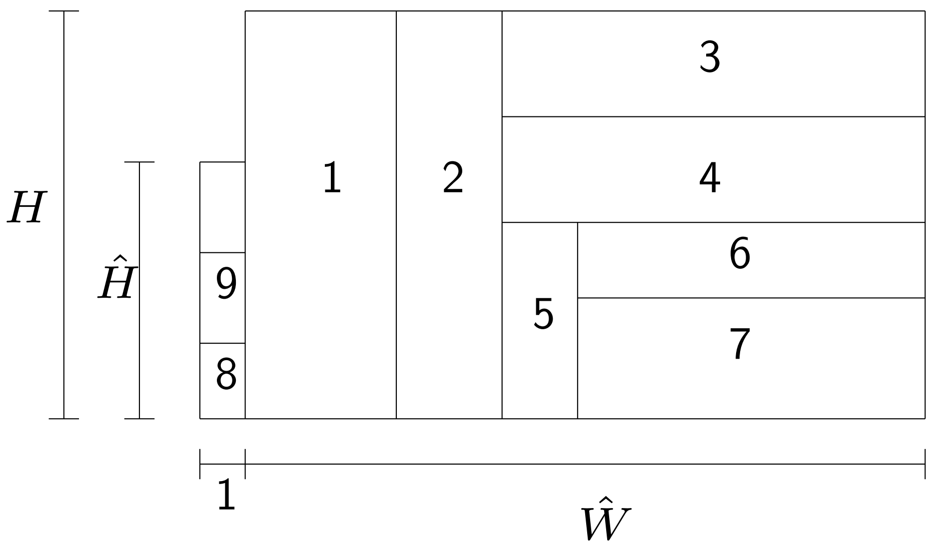
- **Two fast heuristics** embedded in an **interactive algorithm**.
- Description for the more general problem **P2**.
- **First heuristic: Stripes**, derived from the 3-approx algorithm for **P0**:



- the packing depends on the profit per unit area;
- the partial left column is used for the strips.

6. Development of heuristic algorithms: Tiles

- **Second heuristic: Tiles**, totally different philosophy, totally different solutions:



- At each iteration, the best **vertical or horizontal** packing of an item is computed;
- best $\simeq$ minimum waste;
- the partial left column is used for the residual sub-areas.

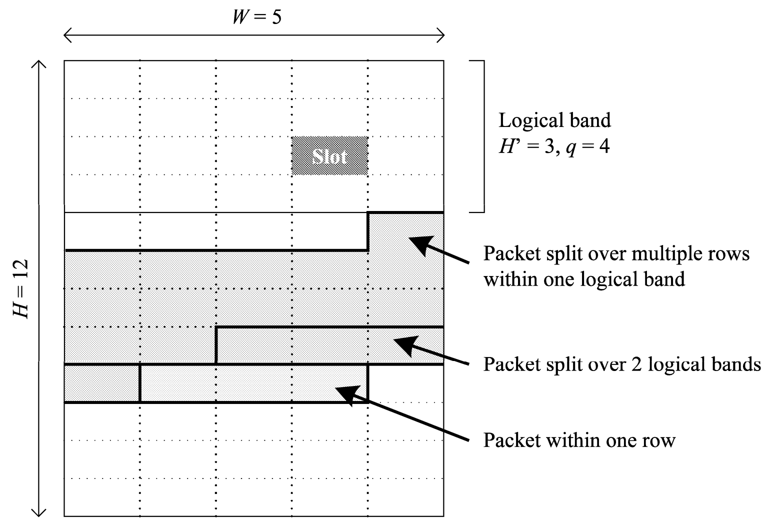
6. Development of heuristic algorithms: Tiles&Stripes

- **Overall heuristic: Tiles&Stripes**:

sort the sub-items according to non-increasing value of their profit per unit area;
initialize the incumbent solution σ to empty;
initialize S to contain all sub-items;
repeat
    define initial tentative values for W and H (comment: usable bin);
    repeat (comment: try to pack the sub-item set S)
        execute Tiles(S) for the current W and H;
        execute Stripes(S) for the current W and H;
        compute the corresponding maps, and let τ be the best feasible solution, if any;
        if a feasible τ has been found **then**
            possibly update σ with τ, and increase the current W and H
        else decrease the current W and H
    until τ includes all sub-items of S **or** limit on number of iterations has been reached;
    if all sub-items of the instance have been allocated **then** terminate;
    if all sub-items of S have been allocated **then** add sub-items to S;
    else remove sub-items from S
until a prefixed maximum number of iterations has been executed.

Back to the real world problems: P3 (Adjacent Permutation Zone)

- Each **data packet** j has an **area** a_j (bytes) and a **profit** p_j (priority).
- The available zone is a $W \times H$ (time $\times$ frequency) rectangle consisting of an **array of slots**:



- Contiguous rows grouped q by q into $H' = H/q$ **logical bands**;
- Matrix E of n columns (one per data packet) and H' rows (one per logical band):

  $e_{ij} = \#$ bytes of data packet j that could be accommodated into a single slot of logical band i;
- **data packets** allocated to **contiguous slots** in row-wise manner, possibly over multiple rows;
- if data packet j is allocated to one logical band, say i, then the number of slots needed is $\frac{a_j}{e_{ij}}$;

  if the allocation spans over a set of contiguous logical bands, then it is $\frac{a_j}{\min\{e_{ij}\}}$.

Theoretical analysis and heuristics for Problem P3

- Packing a maximum profit subset of packets is a **strongly $\mathcal{NP}$-hard** problem.
- Proof: transformation from the one-dimensional bin packing problem.
- Preliminary empirical analysis:

  The optimal solutions "very rarely" split packets between consecutive bands;
- reasonable because when splitting occurs the less favorable e_{ij} is used ($\dfrac{a_j}{\min\{e_{ij}\}}$);
- splitting only occurs for "large" high-priority packets that do not fit alone into a unique logical band.
- **Two-phase algorithm:**

  1. pack the "large" high-priority packets in a greedy way;

  2. Pack the remaining packets without splitting:

  – **Packing without splitting** can be reformulated as a **Generalized Assignment Problem**.

  – Solved by adapting heuristics for the GAP.

7. Implementation and experimental evaluation on realistic scenarios.

- All algorithms have been coded in C

  and run on a 2.40 GHz, CORE 2 DUO E6600 Desktop, running under Linux.

- The computer networking group (University of Pisa) and the Nokia Siemens laboratory implemented a realistic simulator for both kinds of model:

- mix of data and voice users;

- higher priority to packets directed to users with an ongoing voice conversation;

- different packet sizes for data and voice traffic;

- different ratios between the number of users with data traffic and those with voice conversations.

7. Computational experiments, Probl. P2 (Distributed Permutation Zone)

- More than 90,000 instances representing different scenarios of transmission.
- Computing times in CPU milliseconds.

					Optimality				Time (ms)	
	W	H	n	# inst.	# pot.	# opt.	# good	Avg. $z = U$	Avg. T	Max T
B1	17	10	$[1, 13]$	23,040	23,040	22,114	22,846	0.9971	0.038	0.410
B2	17	30	$[1, 15]$	23,040	23,040	21,840	23,014	0.9977	0.078	0.540
C1	17	10	$[1, 15]$	23,210	10,158	8,340	13,719	0.9241	0.085	0.550
C2	17	30	$[1, 26]$	23,317	2,512	1,788	4,544	0.8378	0.196	0.960

- # pot. = instances for which
  (Total area) + (map space for a solution with one rectangle per packet) $\leq WH$;
- U = simple (and very optimistic) upper bound on the maximum area that can be packed;
- # opt. = instances for which $z = U$;
- # good = instances for which the ratio z/maximum packable area ≥ 0.9

7. Computational experiments, Probl. P3 (Adjacent Permutation Zone)

- 54,000 instances representing different scenarios of transmission.
- Computing times in CPU milliseconds.

					Optimality				Time (ms)	
	W	H	n	# inst.	# pot.	# opt.	# good	Avg. $z = U$	Avg. T	Max T
B-1	8	48	$[12, 45]$	9,000	9,000	8,204	9,000	0.9994	0.067	0.430
B-4	8	48	$[12, 47]$	9,000	9,000	8,271	8,999	0.9995	0.062	0.460
B-both	8	48	$[12, 47]$	9,000	9,000	8,210	9,000	0.9994	0.064	0.550
U-1	8	48	$[10, 47]$	9,000	8,091	5,321	8,164	0.9790	0.051	0.220
U-4	8	48	$[10, 60]$	9,000	8,003	5,070	8,176	0.9793	0.054	0.220
U-both	8	48	$[20, 77]$	9,000	7,057	2,218	8,197	0.9749	0.158	0.420

- U = solution to a 0-1 knapsack problem relaxation of P3:

  profits = priorities, weights $= \dfrac{a_j}{\min\{e_{ij}\}}$, capacity $= WH$;

- # pot. = instances for which all packets are in the knapsack solution;
- # opt. = instances for which $z = U$;
- # good = instances for which the ratio z/maximum packable area ≥ 0.9

Conclusions

- we have considered real-world packing problems arising in wireless telecommunications, and especially in orthogonal frequency division multiple access (*OFDMA*);
- these real-world packing problems are challenging per se BUT they become even more difficult because of technological constraints which require to solve them within one millisecond.
- we have defined a clean and easy-to-state packing problem (**P0**) that is the core of some of these problems;
- we have proved the complexity status of P0, and we have defined an approximation algorithm with worst-case guarantee;
- we have derived fast and efficient heuristics for the real-world problems;

Thank you for your attention