

# A Facility Location Problem in VLSI Design

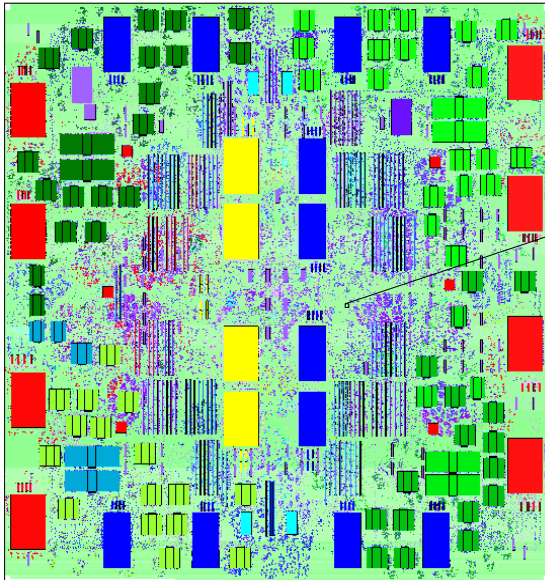
Stephan Held

Research Institute for Discrete Mathematics  
University of Bonn

NATO Summer School in Combinatorial Optimization  
June 2006

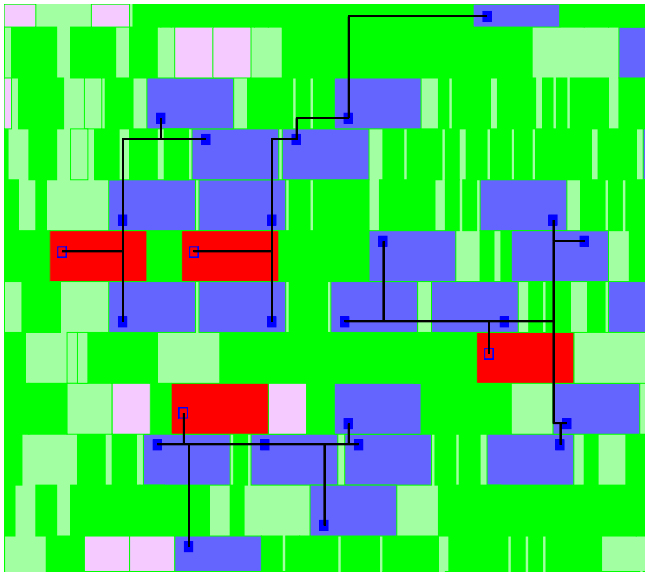
- 1 Motivation
- 2 Problem Statement
- 3 Complexity Results
- 4 Lower Bounds
- 5 Approximation Algorithms
  - Algorithm A – Based on Spanning Tree
  - Algorithm B – Based on Steiner Tree
  - Algorithm C – Based on TSP

# VLSI Design: Distributing a signal to several terminals



Zoom by factor 400  
(1/160000 of chip area)

# VLSI Design: Distributing a signal to several terminals



blue: terminals

red: facilities

# Problem Statement

Instance:

- metric space  $(V, c)$ ,
- finite set  $\mathcal{D} \subseteq V$  (terminals/clients),
- demands  $d : \mathcal{D} \rightarrow \mathbb{R}_+$ ,
- facility opening cost  $f \in \mathbb{R}_+$ ,
- capacity  $u \in \mathbb{R}_+$ .

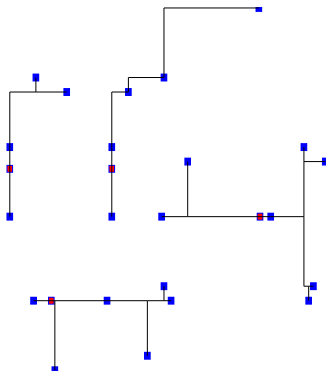
Find a partition  $\mathcal{D} = D_1 \dot{\cup} \dots \dot{\cup} D_k$  and Steiner trees  $T_i$  for  $D_i$  ( $i = 1, \dots, k$ ) with

$$c(E(T_i)) + d(D_i) \leq u$$

for  $i = 1, \dots, k$  such that

$$\sum_{i=1}^k c(E(T_i)) + kf$$

is minimum.



# Related Work

## Steiner Tree Problem:

- Robins and Zelikovsky 2000

## Bin Packing Problem:

- Simchi-Levi, 1994

## Soft Capacitated Facility Location Problems:

- Chudak, Shmoys 1999
- Mahdian, Ye, Zhang, 2003

## Clustering Problems:

- Even, Garg, Könemann, Ravi, Sinha, 2003
- Toth, Viga, 2002

# Complexity Results

(All the following results are by Maßberg and Vygen 2005)

## Proposition

*There is no  $(2 - \epsilon)$ -approximation algorithm (for any  $\epsilon > 0$ ) for any class of metrics where the Steiner tree problem cannot be solved exactly in polynomial time.*

# Complexity Results

(All the following results are by Maßberg and Vygen 2005)

## Proposition

*There is no  $(2 - \epsilon)$ -approximation algorithm (for any  $\epsilon > 0$ ) for any class of metrics where the Steiner tree problem cannot be solved exactly in polynomial time.*

## Proof.

Is there a Steiner tree with terminals  $S$  and length  $\leq k$ ?

# Complexity Results

(All the following results are by Maßberg and Vygen 2005)

## Proposition

*There is no  $(2 - \epsilon)$ -approximation algorithm (for any  $\epsilon > 0$ ) for any class of metrics where the Steiner tree problem cannot be solved exactly in polynomial time.*

## Proof.

Is there a Steiner tree with terminals  $S$  and length  $\leq k$ ?

Set  $d(s) = 0, \forall s \in S, u = k$  and  $f = k \frac{2-\epsilon}{\epsilon}$ .

# Complexity Results

(All the following results are by Maßberg and Vygen 2005)

## Proposition

*There is no  $(2 - \epsilon)$ -approximation algorithm (for any  $\epsilon > 0$ ) for any class of metrics where the Steiner tree problem cannot be solved exactly in polynomial time.*

## Proof.

Is there a Steiner tree with terminals  $S$  and length  $\leq k$ ?

Set  $d(s) = 0, \forall s \in S, u = k$  and  $f = k \frac{2-\epsilon}{\epsilon}$ .

An  $(2 - \epsilon)$ -approximation algorithm computes a solution consisting of a single facility

$\Leftrightarrow$  There is a Steiner tree of length  $\leq k$ .



## Proposition

- *There is no  $(1.5 - \epsilon)$ -approximation algorithm (for any  $\epsilon > 0$ ) unless  $P = NP$  (Transformation from Partition Problem).*
- *There is a 2-approximation algorithm for geometric instances (similar to Arora's approximation scheme for the TSP). However, this is not practically efficient.*

## Lower bound: spanning forests

Let  $F_1$  be a minimum spanning tree for  $(\mathcal{D}, c)$ .

Let  $e_1, \dots, e_{n-1}$  be the edges of  $F_1$  so that  $c(e_1) \geq \dots \geq c(e_{n-1})$ .

Set  $F_k := F_{k-1} \setminus \{e_{k-1}\}$  for  $k = 2, \dots, n$ .

### Lemma

*$F_k$  is a minimum weight spanning forest in  $(\mathcal{D}, c)$  with exactly  $k$  components.*

### Proof.

By induction on  $k$ . Trivial for  $k = 1$ . Let  $k > 1$ .

Let  $F^*$  be a minimum weight  $k$ -spanning forest.

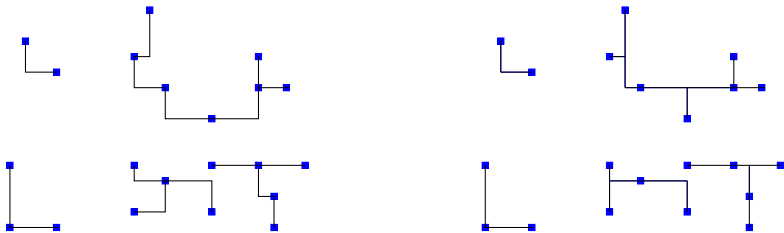
Let  $e \in F_{k-1}$  such that  $F^* \cup \{e\}$  is a forest. Then

$$c(F_k) + c(e_{k-1}) = c(F_{k-1}) \leq c(F^*) + c(e) \leq c(F^*) + c(e_{k-1}).$$



# Lower bound: Steiner forests

A  $k$ -Steiner forest is a forest  $F$  with  $\mathcal{D} \subseteq V(F)$  and exactly  $k$  components.



## Lemma

$\frac{1}{\alpha} c(F_k)$  is a lower bound for the cost of a minimum weight  $k$ -Steiner forest, where  $\alpha$  is the Steiner ratio.

## Lower bound: number of facilities

Let  $t'$  be the smallest integer such that

$$\frac{1}{\alpha}c(F_{t'}) + d(\mathcal{D}) \leq t' \cdot u$$

### Lemma

*$t'$  is a lower bound for the number of facilities of any solution.*

Let  $t''$  be an integer in  $\{t', \dots, n\}$  minimizing

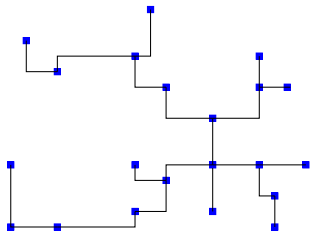
$$\frac{1}{\alpha}c(F_{t''}) + t'' \cdot f.$$

### Theorem

*$\frac{1}{\alpha}c(F_{t''}) + t'' \cdot f$  is a lower bound for the cost of an optimal solution.*

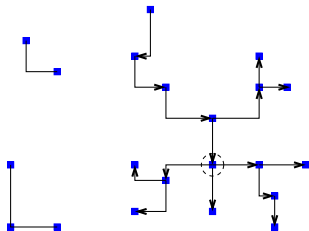
# Algorithm A

- 1 Compute a minimum spanning tree on  $(\mathcal{D}, c)$ .



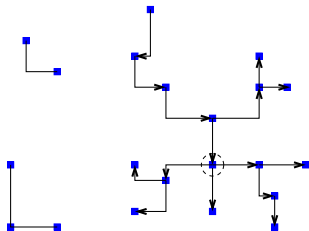
# Algorithm A

- 1 Compute a minimum spanning tree on  $(\mathcal{D}, c)$ .
- 2 Compute  $t''$  and spanning forest  $F_{t''}$  as above.



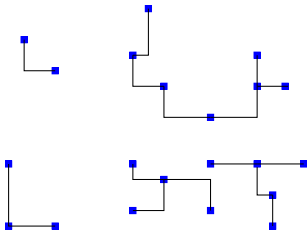
# Algorithm A

- 1 Compute a minimum spanning tree on  $(\mathcal{D}, c)$ .
- 2 Compute  $t''$  and spanning forest  $F_{t''}$  as above.
- 3 Split up overloaded components by a bin packing approach.



# Algorithm A

- 1 Compute a minimum spanning tree on  $(\mathcal{D}, c)$ .
- 2 Compute  $t''$  and spanning forest  $F_{t''}$  as above.
- 3 Split up overloaded components by a bin packing approach.



It can be guaranteed that for each new component at least  $\frac{u}{2}$  of load will be removed from the initial forest.

# Analysis of Algorithm A

**Recall:**  $\frac{1}{\alpha}c(F_{t''}) + t'' \cdot f$  is a lower bound for the optimum.

We set  $L_r := \frac{1}{\alpha}c(F_{t''})$  and  $L_f := t'' \cdot f$ .

**Observe:**  $L_r + d(\mathcal{D}) \leq \frac{u}{f}L_f$ .

The cost of the final solution is at most

$$\begin{aligned}c(F_{t''}) + t''f + \frac{2}{u}\left(c(F_{t''}) + d(\mathcal{D})\right)f \\&= \alpha L_r + L_f + \frac{2f}{u}(\alpha L_r + d(\mathcal{D})) \\&\leq \alpha L_r + L_f + 2\alpha L_f\end{aligned}$$

## Theorem

*Algorithm A is a  $(2\alpha + 1)$ -approximation algorithm.*

## Algorithm B

Define metric  $c'$  by  $c'(v, w) := \min\{c(v, w), \frac{uf}{u+2f}\}$ .

- 1 Compute a Steiner tree  $F$  for  $\mathcal{D}$  in  $(V, c')$  with some  $\beta$ -approximation algorithm.
- 2 Remove all edges  $e$  of  $F$  with  $c(e) \geq \frac{uf}{u+2f}$ .
- 3 Split up overloaded components of the remaining forest as in algorithm A.

### Theorem

*Algorithm B has performance ratio  $3\beta$ .*

Using the Robins-Zelikovsky Steiner tree approximation algorithm we get a 4.648-approximation algorithm.

With a more careful analysis of the Robins-Zelikovsky algorithm we can get a 4.099-approximation algorithm in  $O(n^{2^{10000}})$  time.

# Algorithm C

Define metric  $c''$  by  $c''(v, w) := \min\{c(v, w), \frac{uf}{u+f}\}$

- 1 Compute a tour  $F$  for  $\mathcal{D}$  in  $(V, c'')$  with some  $\gamma$ -approximation algorithm.
- 2 Remove the longest edge of  $F$ .
- 3 Remove all edges  $e$  of  $F$  with  $c(e) \geq \frac{uf}{u+f}$ .
- 4 Split up overloaded components of the remaining forest as in algorithm A.

## Theorem

*Algorithm C has performance ratio  $3\gamma$ .*

Using Christofides' TSP approximation algorithm we get a 4.5-approximation algorithm in  $O(n^3)$  time.

# Comparison of the three approximation algorithms

- Algorithm A computes a minimum spanning tree.
- Algorithm B calls the Robins-Zelikovsky algorithm.
- Algorithm C calls Christofides' algorithm.
- Then each algorithm deletes expensive edges and splits up overloaded components.

algorithm	metric	perf.guar.	runtime
A	$(\mathbb{R}^2, \ell_1)$	4	$O(n \log n)$
A	general	5	$O(n^2)$
B	general	4.099	$O(n^{2^{10000}})$
C	general	4.5	$O(n^3)$

# Experimental Results

Algorithm A on six real-world instances:

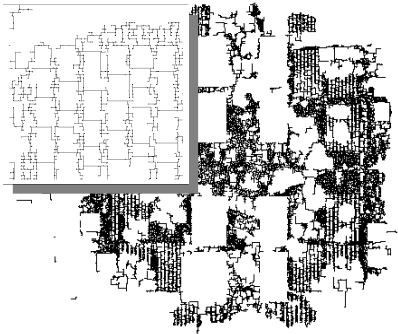
	inst1	inst2	inst3	inst4	inst5	inst6
# terminals	3675	17140	45606	54831	109224	119461
MST length	13.72	60.35	134.24	183.37	260.36	314.48
$t'$	117	638	1475	2051	3116	3998
$L_r$	8.21	31.68	63.73	102.80	135.32	181.45
$L_r + L_f$	23.07	112.70	251.06	363.28	531.05	689.19
# facilities	161	947	2171	2922	4156	5525
service cost	12.08	54.23	101.57	159.93	234.34	279.93
total cost	32.52	174.50	377.29	531.03	762.15	981.61
gap (factor)	1.41	1.55	1.59	1.46	1.44	1.42

# Reduction of power consumption

Algorithm A on four chips, compared to the previously used heuristic:

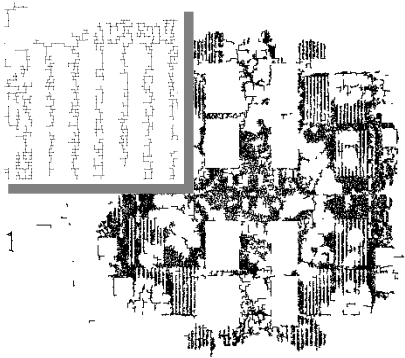
chip	Jens	Katrin	Bert	Alex
technology	180nm	130nm	130nm	130nm
# clock etrees	1	3	69	195
total # sinks	3805	137265	40298	189341
largest instance	375	119461	16260	35305
power (W, old)	0.100	0.329	0.306	2.097
power (W, new)	0.088	0.287	0.283	1.946
difference	-11.1%	-12.8%	-7.5%	-7.2%

Greedy Strategy



Power consumption: 2.17 W

Algorithm A



Power consumption 1.89 W

Typically 10%-15% power reduction compared to greedy strategy.