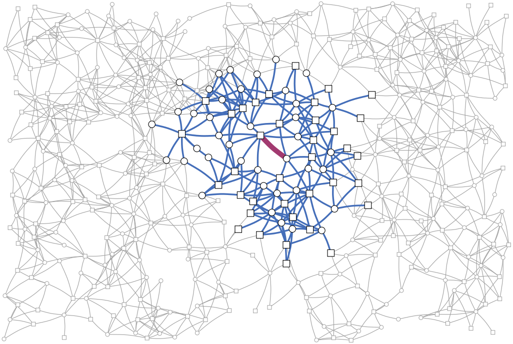
Approximating max-min linear programs with local algorithms

Patrik Floréen,
Petteri Kaski,
Topi Musto,
Jukka Suomela

HIIT,
University of Helsinki,
Finland

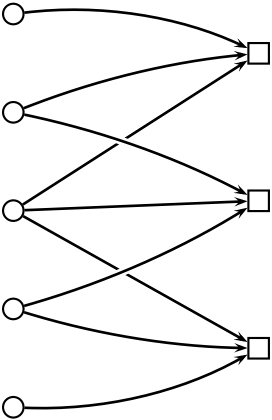


IPDPS
17 April 2008

Max-min linear programs: Example

Example: Fair bandwidth allocation in a communication network

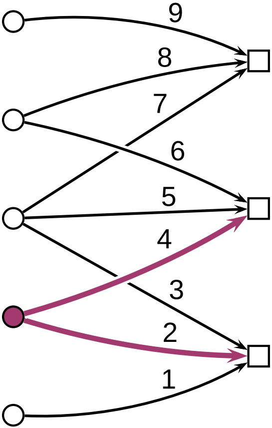
- circle = customer
- square = access point
- edge = network connection



Max-min linear programs: Example

Example: Allocate a fair share of bandwidth for each customer

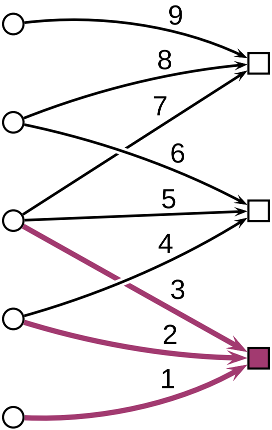
$$\text{maximise} \quad \min \Big\{ x_1,\ \underline{x_2 + x_4},\ x_3 + x_5 + x_7,\ x_6 + x_8,\ x_9 \Big\}$$



Max-min linear programs: Example

Example: Allocate a fair share of bandwidth for each customer; each access point has a limited capacity

$$\begin{aligned} \text{maximise} \quad & \min \{ \\ & \quad x_1,\ x_2 + x_4, \\ & \quad x_3 + x_5 + x_7, \\ & \quad x_6 + x_8,\ x_9 \\ & \} \\ \text{subject to} \quad & x_1 + x_2 + x_3 \le 1, \\ & x_4 + x_5 + x_6 \le 1, \\ & x_7 + x_8 + x_9 \le 1, \\ & x_1, x_2, \ldots, x_9 \ge 0 \end{aligned}$$

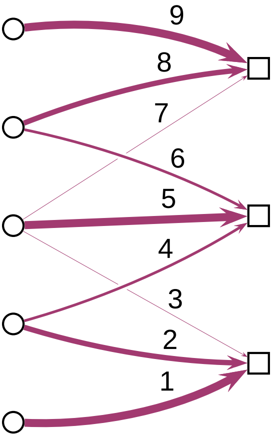


Max-min linear programs: Example

Example: Allocate a fair share of bandwidth for each customer; each access point has a limited capacity

An optimal solution:

$$
\begin{aligned}
x_1 = x_5 = x_9 &= 3/5, \\
x_2 = x_8 &= 2/5, \\
x_4 = x_6 &= 1/5, \\
x_3 = x_7 &= 0
\end{aligned}
$$



Max-min linear programs: Definition

Objective:

$$\begin{array}{lrcll} \text{maximise} & \min\limits_{k \in K} \sum_{v \in V} c_{kv} x_v & & & \\ \text{subject to} & \sum_{v \in V} a_{iv} x_v & \leq & 1 & \forall\, i \in I, \\ & x_v & \geq & 0 & \forall\, v \in V \end{array}$$

Idea:

- One unit of activity by *agent* $v \in V$ benefits *party* $k \in K$ by $c_{kv} \geq 0$ units and consumes $a_{iv} \geq 0$ units of *resource* $i \in I$
- Objective: set the activities to provide a *fair share of benefit* for each party

Max-min linear programs: Definition

Let $A, c, c_k \geq 0$

In matrix notation:

$$\begin{aligned} \text{maximise} \quad & \min_{k \in K} c_k x \\ \text{subject to} \quad & Ax \leq \mathbf{1}, \\ & x \geq \mathbf{0} \end{aligned}$$

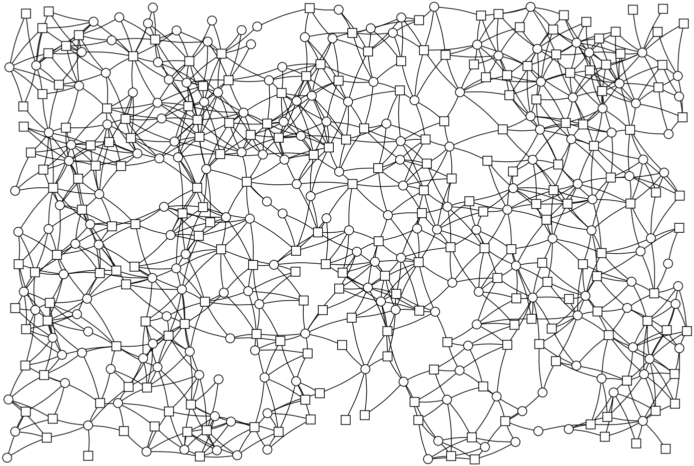
Generalisation of packing LP:

$$\begin{aligned} \text{maximise} \quad & cx \\ \text{subject to} \quad & Ax \leq \mathbf{1}, \\ & x \geq \mathbf{0} \end{aligned}$$

Local algorithms

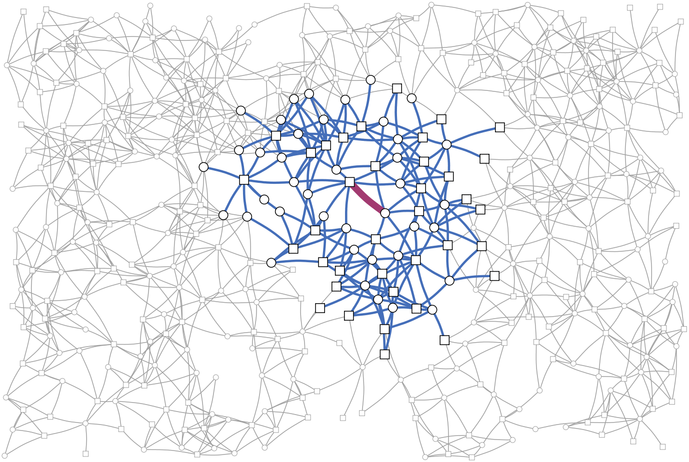
What about large networks?

What if there are frequent changes in network topology?



Local algorithms

Could we perhaps use solely *local* information to find a *provably near-optimal* solution to the global problem?



Local algorithms

Definition: (e.g., Naor and Stockmeyer 1995)

- Distributed algorithm
- Output of a node is a function of input within its *constant-radius neighbourhood*

Our focus:

- Problems where the size of input and output per node is bounded by a constant

Here *constant* = does not depend on input,
in particular, does not depend on the number of nodes
(but may depend on desired approximation ratio, etc.)

Local algorithms

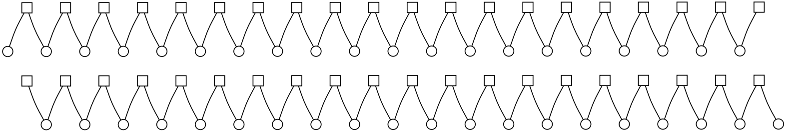
Advantages of a local algorithm:

- Space and time complexity is constant per node
- Distributed constant time (even in an infinite network)
- Topology change affects a constant-size part only
- Simple linear-time centralised algorithm
- In some cases randomised, approximate sublinear-time algorithms (Parnas and Ron 2007)

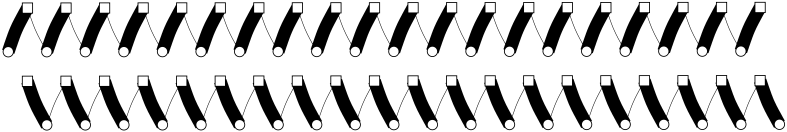
But can we design a local algorithm for max-min LPs?

Challenges of locality

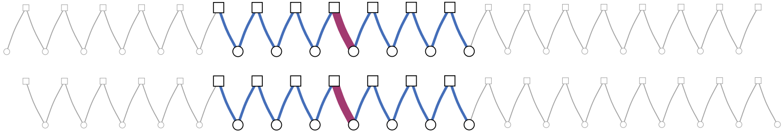
Two instances of the bandwidth allocation problem:



Different optimal solutions:

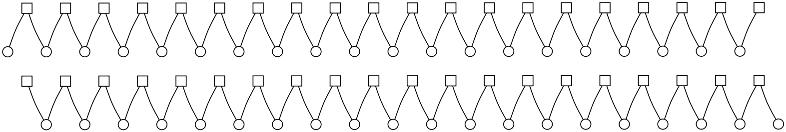


. . . but identical local neighbourhoods:

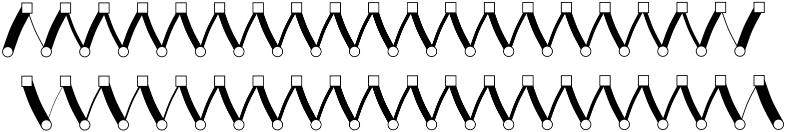


Challenges of locality

Two instances of the bandwidth allocation problem:



Near-optimal solutions:



- Here we can make the same decisions in parts where local neighbourhoods are identical
- Can we generalise this idea to arbitrary instances?

Old results: approximability

Yes, there are local approximation algorithms
for max-min linear programs

"Safe algorithm": node v chooses

$$x_v = \min_{i \,:\, a_{iv} > 0} \frac{1}{a_{iv} \left|\{u : a_{iu} > 0\}\right|}$$

(Papadimitriou and Yannakakis 1993)

This is a factor Δ_I^V approximation where
$\Delta_I^V =$ maximum number of variables in a constraint

Uses information only in radius 1 neighbourhood of v
— a better approximation ratio with a larger radius?

New results: inapproximability

The safe algorithm is factor Δ_I^V approximation

In general, we cannot have a much better approximation ratio:

Theorem

There is no local algorithm for max-min LP with approximation ratio less than

$$\frac{\Delta_I^V + 1}{2} - \frac{1}{2\Delta_K^V - 2}$$

- $\Delta_I^V =$ maximum number of variables in a constraint
- $\Delta_K^V =$ maximum number of variables that benefit a party

Upcoming results: inapproximability

The safe algorithm is factor Δ_I^V approximation

In general, we cannot have a much better approximation ratio (upcoming, tight result):

Theorem

There is no local algorithm for max-min LP with approximation ratio

$$\Delta_I^V \left(1 - \frac{1}{\Delta_K^V}\right)$$

- $\Delta_I^V =$ maximum number of variables in a constraint
- $\Delta_K^V =$ maximum number of variables that benefit a party

New results: approximability

Define *relative growth*

$$\gamma(r) = \max_{v \in V} \frac{|B_{\mathcal{H}}(v, r+1)|}{|B_{\mathcal{H}}(v, r)|}$$

where $B_{\mathcal{H}}(v, r) =$ radius r neighbourhood of v in $\mathcal{H}$

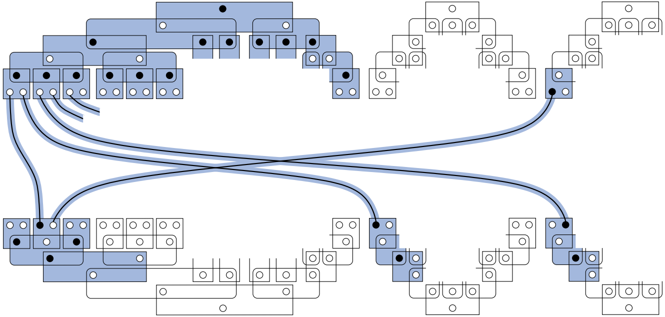
If $\mathcal{H}$ has bounded relative growth, then
better approximation ratios can be achieved:

Theorem

For any R, there is a local algorithm for max-min LP with approximation ratio $\gamma(R-1)\,\gamma(R)$ and local horizon $\Theta(R)$

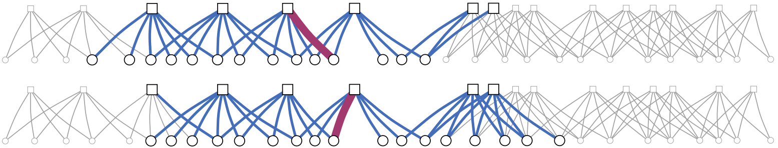
Proof idea: inapproximability

- Construct instance $\mathcal{S}$ with no short cycles
- Apply the supposed approximation algorithm $\mathcal{A}$ to $\mathcal{S}$
- Study the solution; choose a “bad” tree-like area $\mathcal{S}' \subset \mathcal{S}$
- $\mathcal{A}$ has to make the same local decisions in $\mathcal{S}'$, suboptimal

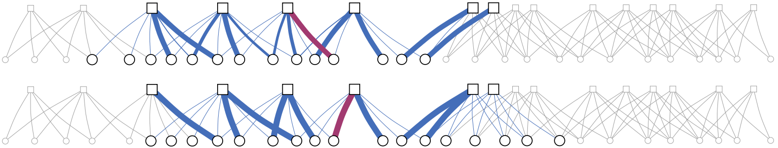


Algorithm idea: approximability

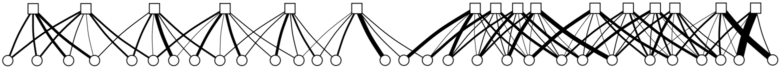
Choose local constant-size subproblems:



Solve them optimally:



Take averages of local solutions, add some slack:



Summary

Max-min linear programs: given $A, c_k \geq 0$,

$$\begin{array}{ll} \text{maximise} & \min_{k \in K} c_k x \\ \text{subject to} & Ax \leq \mathbf{1},\ x \geq \mathbf{0} \end{array}$$

Local algorithms: output is a function of input in a constant-radius neighbourhood

Results:

- Inapproximability results for general graphs
- Approximation algorithm for bounded-growth graphs

http://www.hiit.fi/ada/geru — jukka.suomela@cs.helsinki.fi

References (1)

P. Floréen, P. Kaski, T. Musto, and J. Suomela. Approximating max-min linear programs with local algorithms. In *Proc. 22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS, Miami, FL, USA, April 2008)*, 2008. To appear.

M. Naor and L. Stockmeyer. What can be computed locally? *SIAM Journal on Computing*, 24(6):1259–1277, 1995. [DOI]

C. H. Papadimitriou and M. Yannakakis. Linear programming without the matrix. In *Proc. 25th Annual ACM Symposium on Theory of Computing (STOC, San Diego, CA, USA, May 1993)*, pages 121–129, New York, NY, USA, 1993. ACM Press. [DOI]

M. Parnas and D. Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1–3):183–196, 2007. [DOI]