The Rank Aggregation Problem

David P. Williamson
Cornell University

Universidade Federal de Minas Gerais
December 10, 2012

Outline

- An old problem and one formulation of it
- Some modern-day applications
- Related work in approximation algorithms
- Some computational results
- Conclusion

An old question

- How can the preferences of multiple competing agents be fairly taken into account?
  - Groups deciding where to go to dinner
  - Elections

Rank aggregation

- Input:
  - N candidates
  - K voters giving (partial) preference list of candidates
- Goal:
  - Want single ordering of candidates expressing voters' preferences
  - ???

Ballot

1. Labour
2. Liberal Democrats

Ballot

1. Sinn Fein
2. Labour
3. Liberal Democrats

Ballot

1. Conservative
2. Liberal Democrats
3. Labour

A well-known answer

- Arrow (1950): They can’t.
- Can’t simultaneously have a means of aggregating preferences that has:
  - Non-dictatorship
  - Pareto efficiency (if everyone prefers A to B, then final order should prefer A to B)
  - Independence of irrelevant alternatives (Given two inputs in which A and B are ranked identically by everyone, the two outputs should order A and B the same)

Still…

- As with computational intractability, we still need to do the best we can.
- Why is this any more relevant now than before?

The information age

- Can easily see the preferences of millions (e.g. Netflix Challenge).
- …and those of a few.
- What if the main players are systematically biased in some way?

The Rank Aggregation Problem

- Question raised by Dwork, Kumar, Naor, Sivakumar, "Rank aggregation methods for the web", WWW10, 2001.
  - Q: How can search-engine bias be overcome?
  - A: By combining results from multiple search engines

Sample search: Waterloo

Google

1. Wikipedia: Battle of Waterloo
2. Wikipedia: Waterloo, ON
3. www.city.waterloo.on.ca (City of Waterloo website)
4. www.uwaterloo.ca (University of Waterloo)
5. www.waterlooindustries.com (High performance tool storage)

Yahoo!

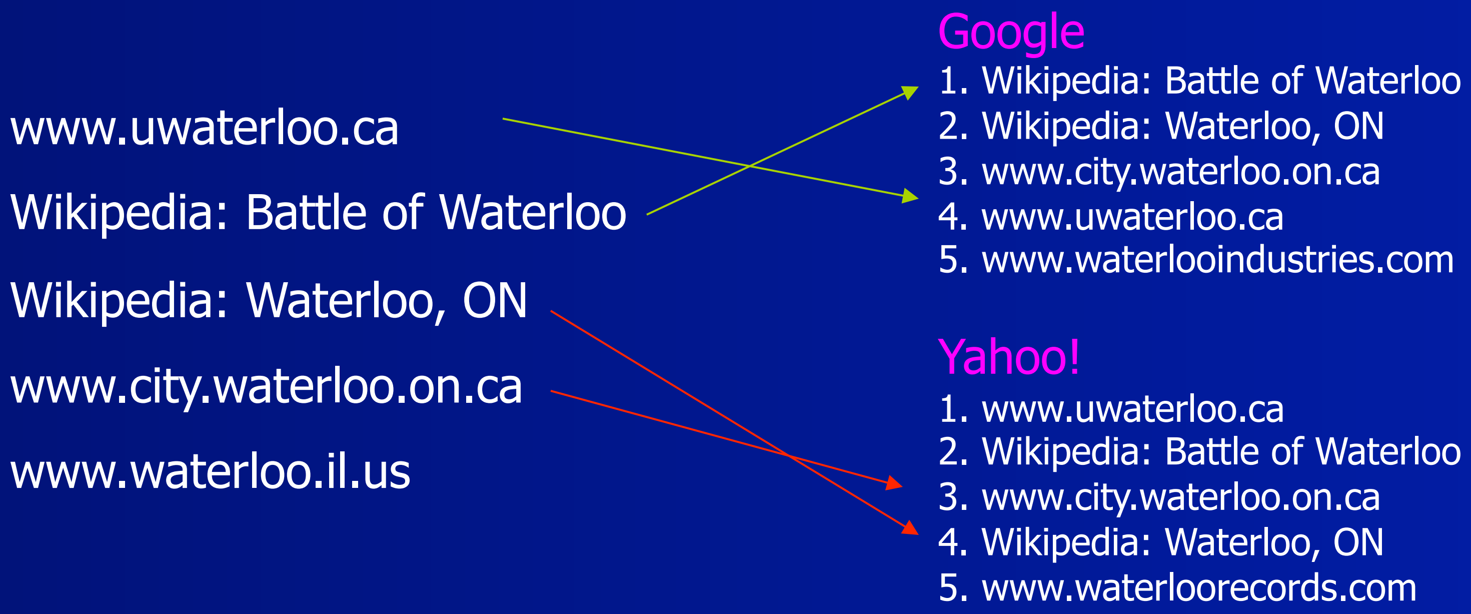
1. www.uwaterloo.ca
2. Wikipedia: Battle of Waterloo
3. www.city.waterloo.on.ca
4. Wikipedia: Waterloo, ON
5. www.waterloorecords.com (Record store in Austin, TX)

MSN

1. Wikipedia: Battle of Waterloo
2. Wikipedia: Waterloo Station (in London)
3. Youtube: Video of ABBA’s “Waterloo”
4. www.waterloorecords.com
5. www.waterloo.il.us (City in Illinois)

Kemeny optimal aggregation

Want to find ordering of all elements that minimizes the total number of pairs "out of order" with respect to all the lists.



www.uwaterloo.ca

Wikipedia: Battle of Waterloo

Wikipedia: Waterloo, ON

www.city.waterloo.on.ca

www.waterloo.il.us

Google

1. Wikipedia: Battle of Waterloo
2. Wikipedia: Waterloo, ON
3. www.city.waterloo.on.ca
4. www.uwaterloo.ca
5. www.waterlooindustries.com

Yahoo!

1. www.uwaterloo.ca
2. Wikipedia: Battle of Waterloo
3. www.city.waterloo.on.ca
4. Wikipedia: Waterloo, ON
5. www.waterloorecords.com

A metric on permutations

Kendall's tau distance K(S,T)

number of pairs (i,j) that S and T disagree on

A	B
B	D
C	A
D	C

number of disagreements: 3 (AB, AD, CD)

- Thus given input top k lists $T_1,\ldots,T_n$, we find permutation S on universe of elements to minimize $K^*(S,T_1,\ldots,T_n) = \Sigma_i K(S,T_i)$ (essentially)
- Yields *extended Condorcet criterion*: if every cand. in A is preferred by some majority to every cand. in B, all of A ranked ahead of all of B.

My home page
Legit.com

Spam.com
Spam.org

But K* NP-hard to compute for 4 or more lists.

How then to compute an aggregation?

- Answer in Dwork et al.: heuristics
- Markov chain techniques: given chain on candidates, compute stationary probs, rank by probs.

Local Kemenization

- Can achieve extended Condorcet by finding S a local min of $K^*(S,T_1,\ldots,T_n)$; i.e. interchanging candidates i and i+1 of S does not decrease score.
- Easy to compute.

Uses

- Internal IBM metasearch engine: Sangam
- IBM experimental *intranet* search engine: iSearch

Fagin, Kumar, McCurley, Novak, Sivakumar, Tomlin, W, “Searching the Workplace Web”, WWW 2003.

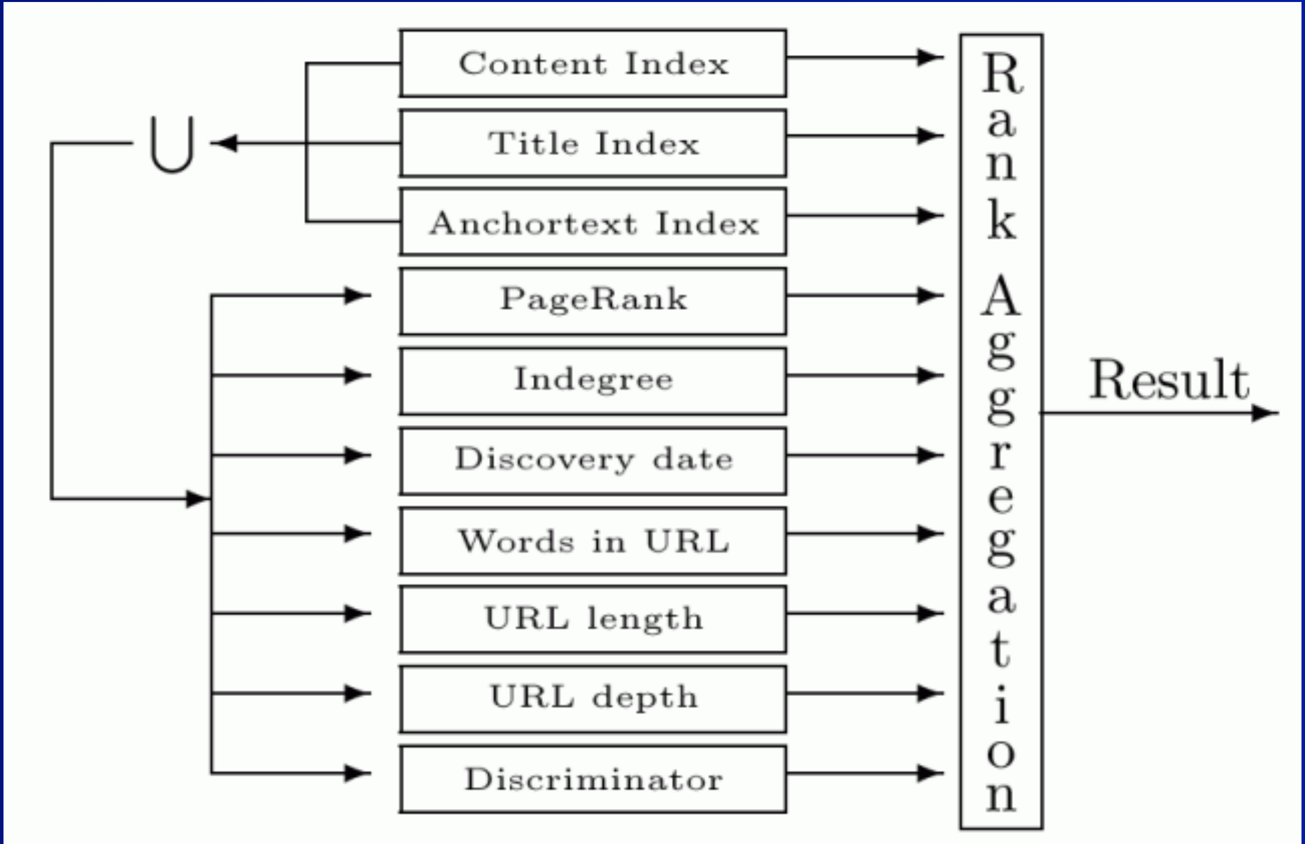
Internet vs. intranet search

- Different social forces at work in content creation
- Different types of queries and results; intranet search closer to ‘home page’ finding
- No spam

eAMT	global print	Travel	Websphere
PBC	e-AMT	Reqcat	ITCS204
HR	jobs	PSM	ITCS300
MTS	TDSP	EPP	vacation planner
ASO	intranet password	redbooks	password
ISSI	global campus	ILC	mobility
Sametime	printers	virus	cell phone
EA2000	human resources	printer	PCF
IDP	ESPP	reserve	BPFJ

iSearch

- Idea: aggregate different ranking heuristics to see what works best for intranet search



Method and results

- Found ground truth, determined "influence" of each ranking heuristic on getting pages into top spot (top 3, top 5, top 10, etc.)
- Best: Anchortext, Titles, PageRank
- Worst: Content, URL Depth, Indegree
- Used Dwork et al. random walk heuristic for aggregation

The Rank Aggregation Problem

- Formulate as a graph problem
- Input:
  - Set of elements V
  - Pairwise information w(i,j),w(j,i)

    w(j,i) = fraction of voters ranking j before i
  - Find a permutation σ that minimizes

    $$\Sigma_{\sigma(i) < \sigma(j)} w(j,i)$$

    (scaled Kemeny aggregation)

Full vs. partial rank aggregation

- *Full* rank aggregation: input permutations are total orders
- *Partial* rank aggregation: otherwise
- Inputs from partial rank aggregation obey triangle inequality:
  - $w(i,j) + w(j,k) \geq w(i,k)$
- Full rank aggregation also obeys probability constraints:
  - $w(i,j) + w(j,i) = 1$

Approximation algorithms

- An α-approximation algorithm is a polynomial-time algorithm that produces a solution of cost at most α times the optimal cost.

Remainder of talk

Approximation algorithms for rank aggregation

- ❑ A very simple 2-approximation algorithm for full rank aggregation
- ❑ Pivoting algorithms
- ❑ A simple, deterministic 2-approximation algorithm for triangle inequality
- ❑ Computational experiments

A simple approximation algorithm

An easy 2-approximation algorithm for full rank aggregation:

choose one of M input permutations at random

probability i is ranked before j =

$$\# \{\pi_m \text{ s.t. } \pi_m(i) < \pi_m(j)\} / M = w(i,j)$$

"cost" if i is ranked before j = w(j,i)

$\Rightarrow$ expected cost for {i,j} :

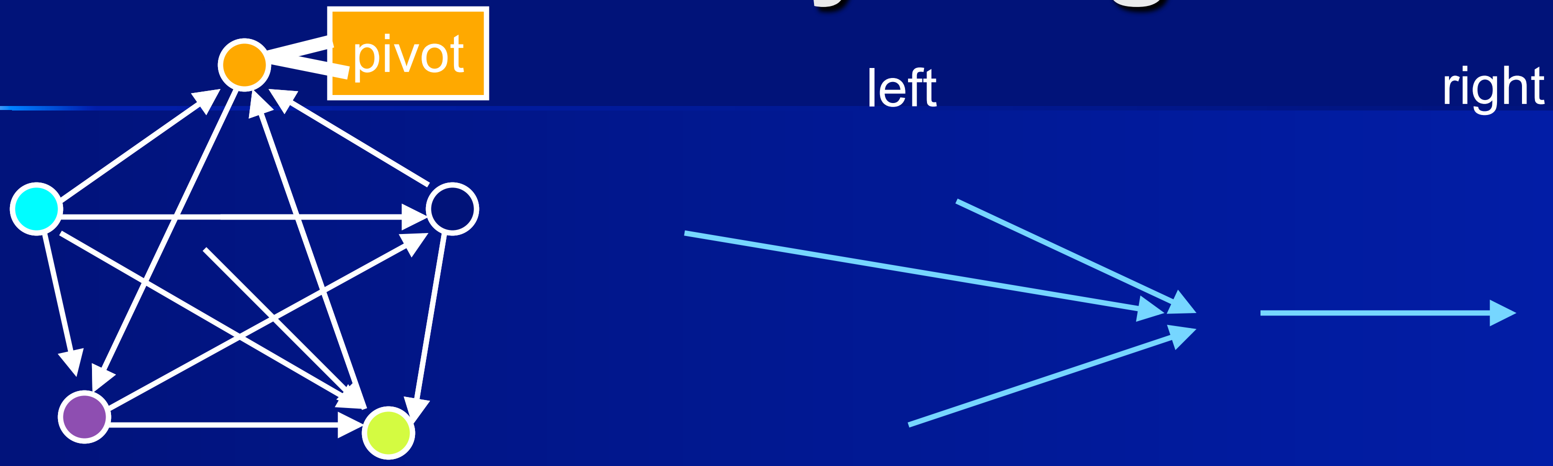
$$2w(i,j)w(j,i) \leq 2 \min \{w(i,j), w(j,i)\}$$

Every feasible ordering has cost for {i,j} at least min {w(i,j), w(j,i)}.

Doing better

- To do better, consider a more general problem in which weights obey triangle inequality and/or probability constraints
  - e.g. problems on tournaments
- Ailon, Charikar, and Newman (STOC 2005) give first constant-factor approximation algorithms for these more general problems.

A Quicksort-style algorithm



- Choose a vertex k as pivot
- Order vertex i

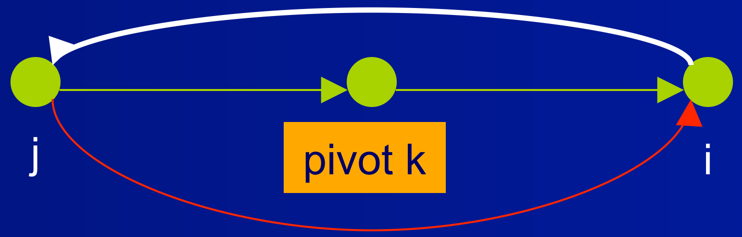
  left of k if (i,k) in A

  right of k if (k,i) in A
- Recurse on left and right

- If graph is weighted, then form a *majority tournament* G=(V,A) that has (i,j) in A if w(i,j) ≥ w(j,i); run algorithm.
- Ailon et al. show that this gives a 3-approximation algorithm for weights obeying triangle inequality
- Van Zuylen & W '07 give a 2-approximation algorithm that chooses the pivot deterministically.

Bounding the cost?

Some arcs in the majority tournament become backward arcs



Observation: backward arcs can be attributed to a pa

"budget" for {i,j}

cost of forward arc = min{w(i,j),w(j,i)} =: w_{ij}
cost of backward arc = max{w(i,j), w(j,i)} =: w_{ij}

Idea: choose pivot carefully, so that the total cost of the backward arcs is not much more than the total budget for these arcs

How to choose a good pivot

Choose pivot minimizing

$$\frac{\text{cost of backward arcs}}{\text{budget of backward arcs}}$$

Thm: If the weights satisfy the triangle inequality, there exists a pivot such that this ratio is at most 2

How to choose a good pivot

There exists a pivot such that

cost of backward arcs ≤ 2 (budget of backward arcs)

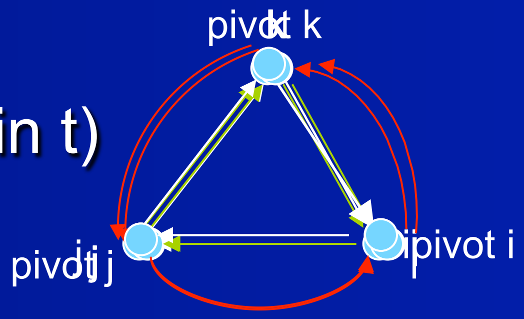
Proof: By averaging argument:

Σ_{pivots} (cost of backward arcs) =

$\Sigma_{\text{directed triangles t}}$ (backward cost of arcs in t)

Σ_{pivots} (budget of backward arcs) =

$\Sigma_{\text{directed triangles t}}$(forward cost of all arcs in t)



How to choose a good pivot

Proof (continued):

Σ_{pivots} (cost of backward arcs) =

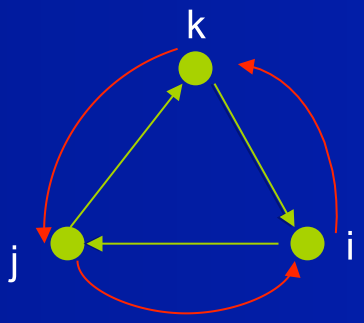
$$\Sigma_{\text{directed triangles } t} \overbrace{(\text{backward cost of arcs in } t)}^{w(t)}$$

Σ_{pivots} (budget of backward arcs) =

$$\Sigma_{\text{directed triangles } t} \underbrace{(\text{forward cost of arcs in } t)}_{w(t)}$$

Not hard to show that

$$w(t) = w(j,i) + w(i,k) + w(k,j)$$
$$\leq w(j,k) + w(k,i) + w(i,j) + w(j,k) + w(k,i) + w(i,j)$$
$$= 2\, w(t)$$



$\Rightarrow$There exists a pivot such that

cost of backward arcs $\leq$ 2 (budget of backward arcs)

Combining the two 2-approximations

Can show that running both the random dictator algorithm and the pivoting algorithm, choosing best solution, gives a 1.6-approximation algorithm for full rank aggregation.

Can be extended to partial rank aggregation

More results

- Ailon, Charikar, Newman ’05 give a randomized LP-rounding 4/3-approximation algorithm for full rank aggregation.
- Ailon ’07 gives 3/2-approximation algorithm for partial rank aggregation.
- Van Zuylen & W ’07 give deterministic variants.
- Kenyon-Mathieu and Schudy ’07 give an approximation scheme for full rank aggregation.

Similar problems

The same sort of pivoting algorithms can be applied to problems in clustering and hierarchical clustering resulting in approximation algorithms with similar performance.

Clustering

- Input:
  - Set of elements V
  - Pairwise information $w^+\{i,j\}$, $w^-\{i,j\}$
  - Assumption: weights satisfy
    - triangle inequality or
    - probability constraints
- Goal:
  - Find a clustering that minimizes

$$\Sigma_{i,j \text{ together}} w^-\{i,j\} + \Sigma_{i,j \text{ separated}} w^+\{i,j\}$$

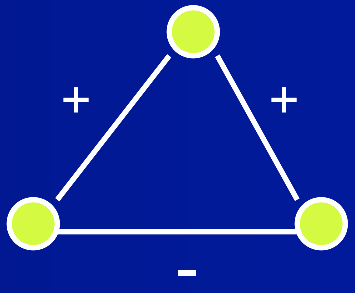
Clustering

"Majority tournament" ⇔
- '+' edge {i,j} if $w^+\{i,j\} \geq w^-\{i,j\}$
- '-'edge {i,j} if $w^-\{i,j\} \geq w^+\{i,j\}$

Pivoting on vertex k:
- If {i,k} is a '+' edge, put i in same cluster as k
- If {i,k} is a '-' edge, separate i from k

Recurse on vertices separated from k

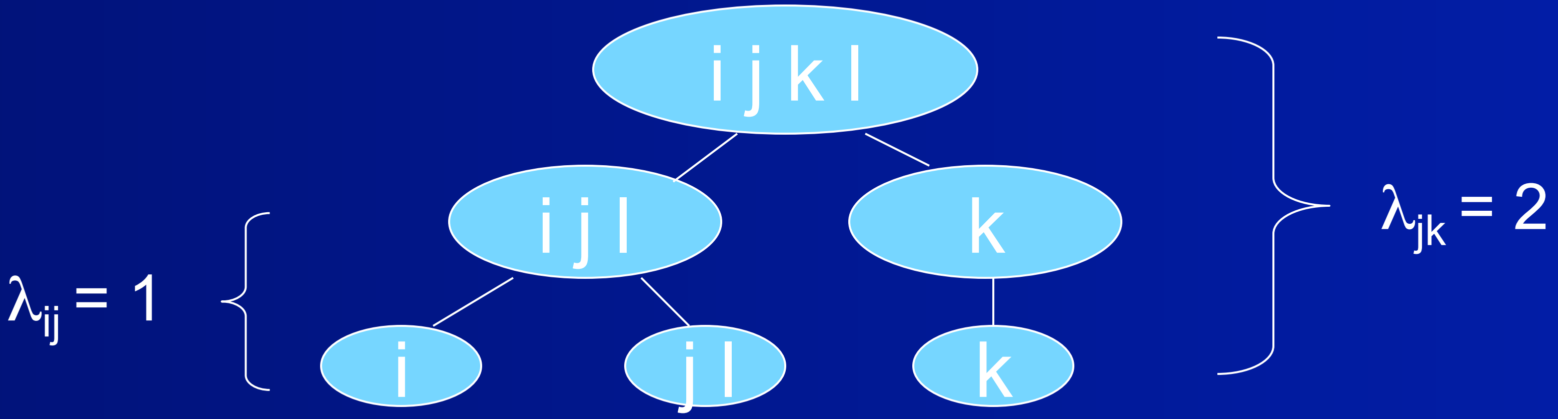
"Directed triangle" ⇔



Hierarchical Clustering

M-level hierarchical clustering :

– M nested clusterings of same set of objects



- Input: pairwise information $D_{ij} \in \{0, \ldots, M\}$
- Goal: Minimize L_1-distance from D: $\Sigma_{i,j} |\lambda_{ij} - D_{ij}|$

Hierarchical Clustering

Hierarchical clustering:

- Construct hierarchical clustering top-down:
    - Use clustering algorithm to get top level clustering
    - Recursively invoke algorithm for each top level cluster

$\Rightarrow$ (M+2)-approximation algorithm (M = # levels)

Matches bound of a more complicated, randomized algorithm of Ailon and Charikar (FOCS ’05)

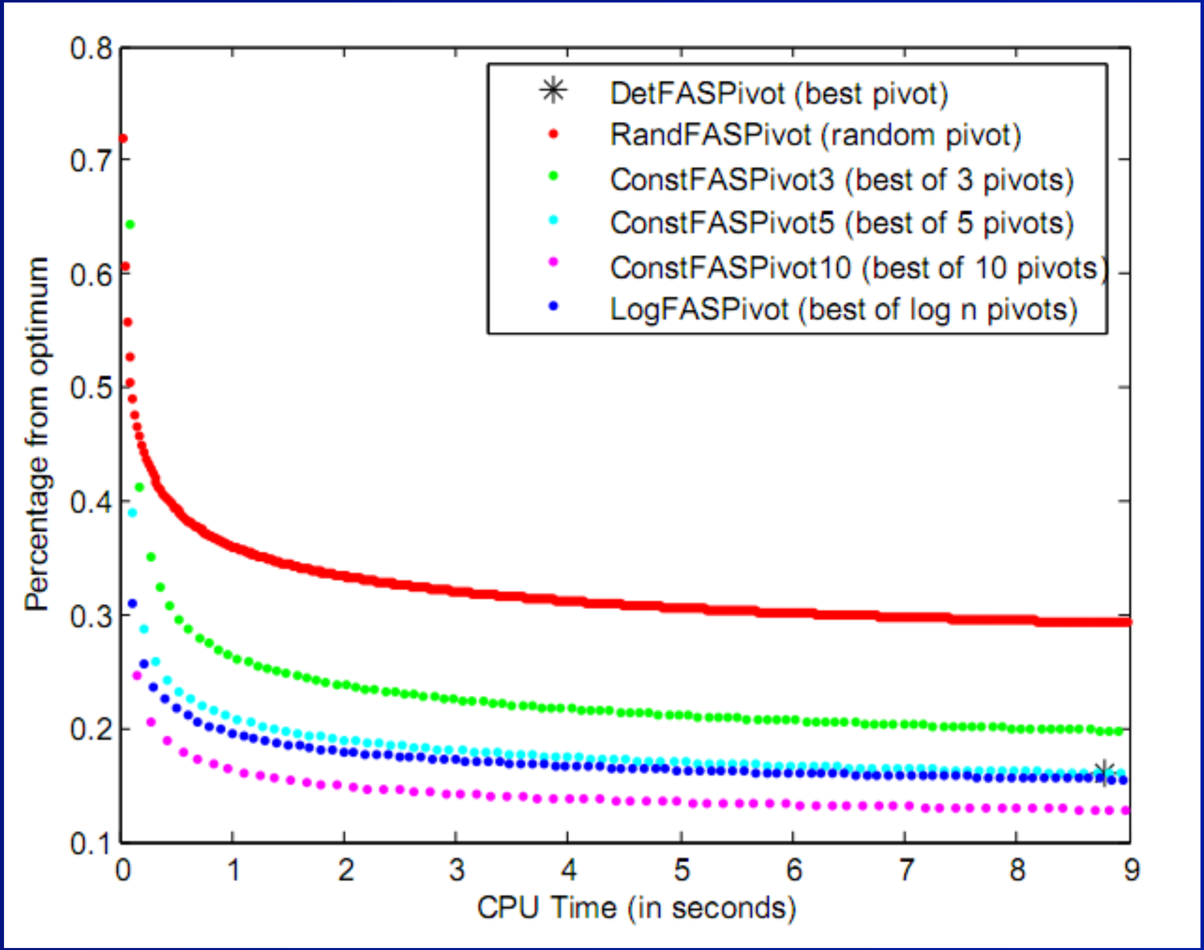
Empirical results

- How well do the ranking algorithms do in practice?
- Two data sets:
  - Repeat of Dwork et al. experiments
    - 37 queries to Ask, Google, MSN, Yahoo!
    - Take top 100 results of each; pages are "same" if canonicalized URLs are same
  - Web Communities Data Set
    - From 9 full rankings of 25 million documents
    - 50 samples of 100 documents, induced 9 rankings of the 100 documents

Pivoting variants

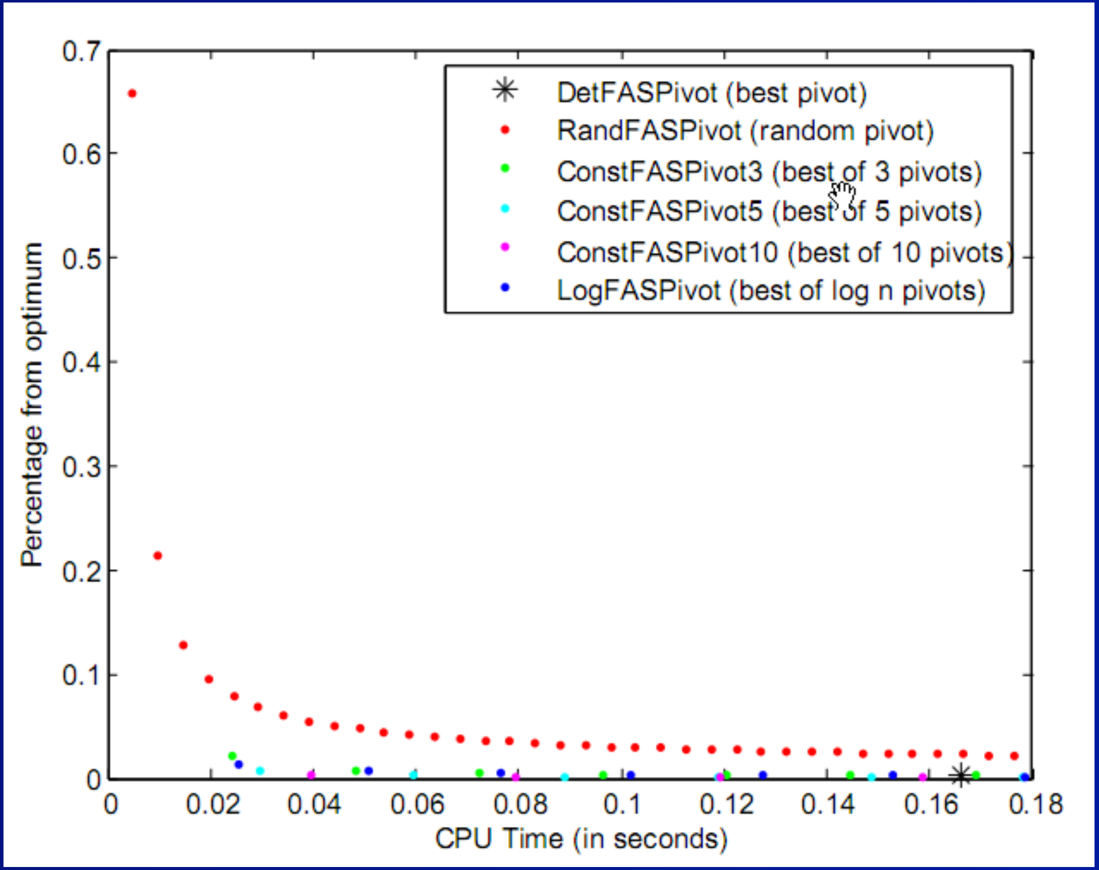
- Deterministic algorithm too slow
- Take K elements at random, use best of K for pivot (using ratio test)

Dwork et al.

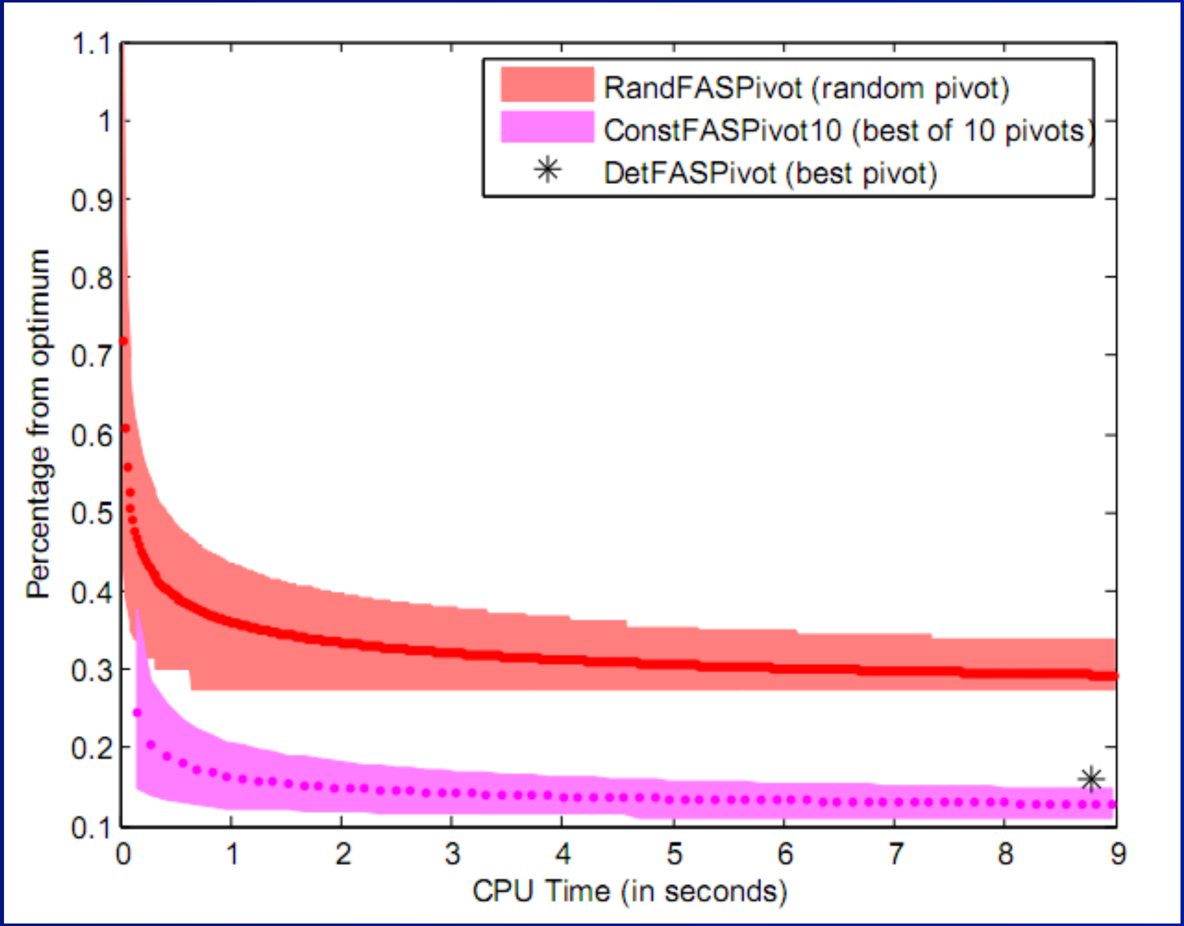


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
Percentage from optimum
0
1
2
3
4
5
6
7
8
9
CPU Time (in seconds)
DetFASPivot (best pivot)
RandFASPivot (random pivot)
ConstFASPivot3 (best of 3 pivots)
ConstFASPivot5 (best of 5 pivots)
ConstFASPivot10 (best of 10 pivots)
LogFASPivot (best of log n pivots)

Web Communities



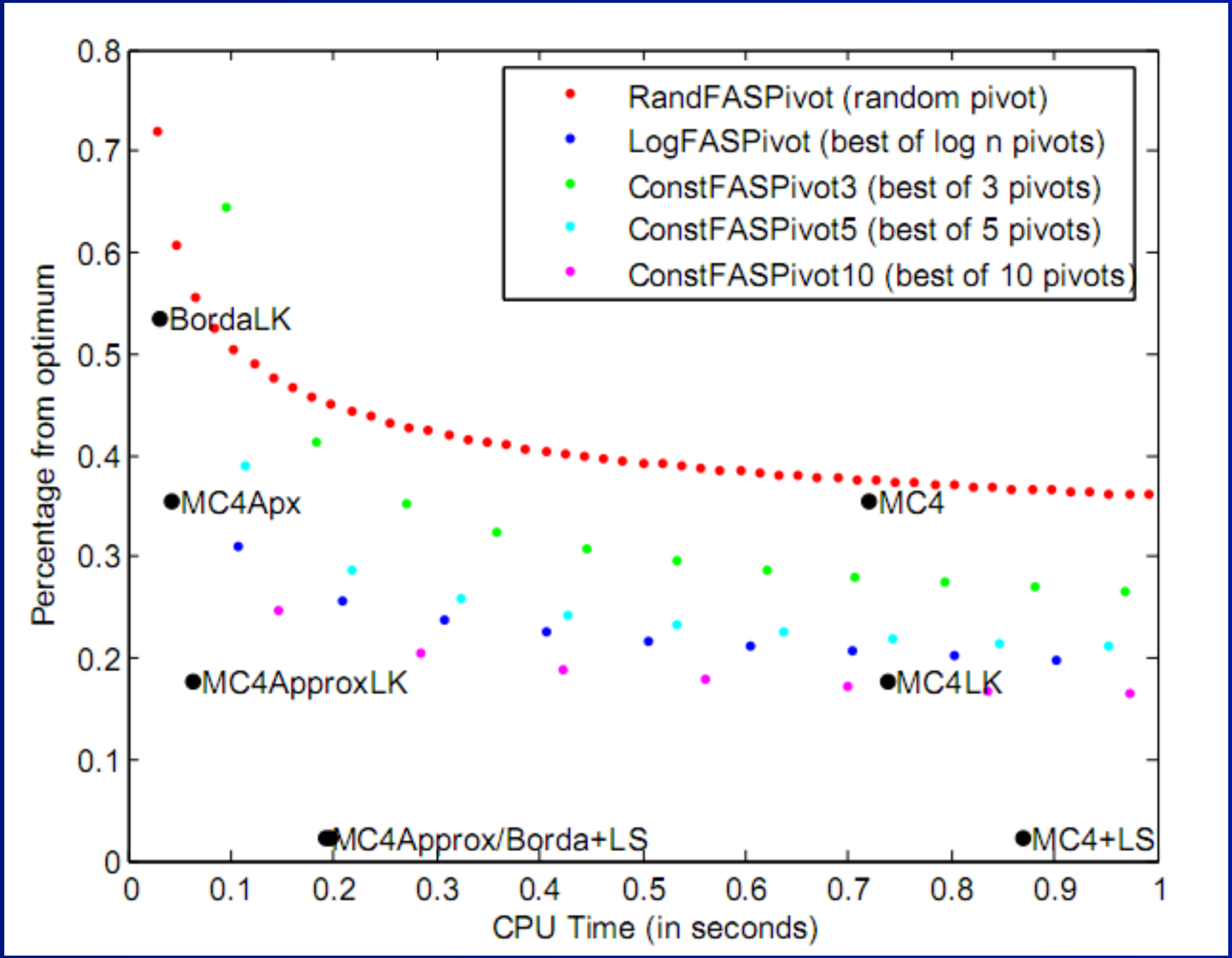
Concentration



Other heuristics

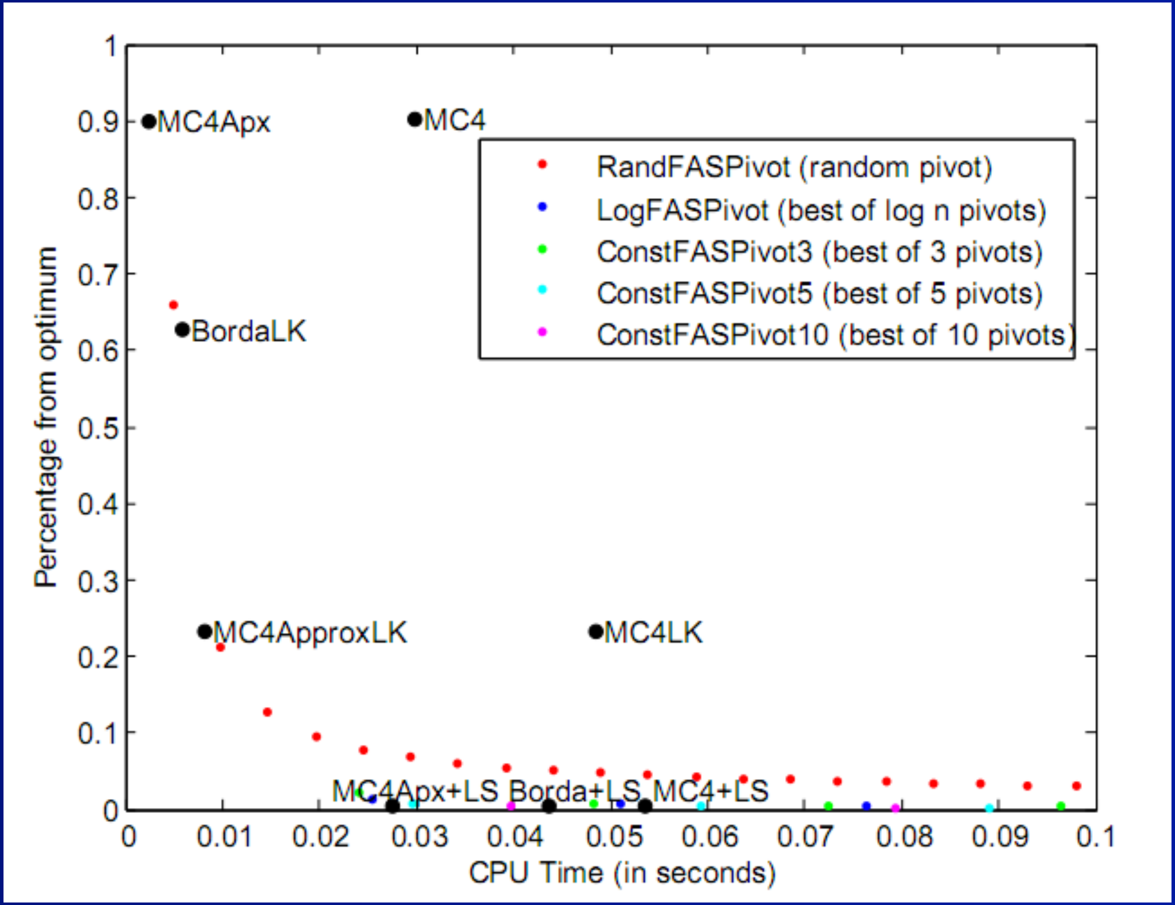
- Borda scoring
  - Sort vertices in ascending order of weighted indegree
- MC4
  - The Dwork et al. Markov Chain heuristic
- Local Kemenization
  - Interchange neighbors to improve overall score
- Local search
  - Move single vertices to improve overall score
- CPLEX LP/IP
  - Most LP solutions integral

Dwork et al.



RandFASPivot (random pivot)
LogFASPivot (best of log n pivots)
ConstFASPivot3 (best of 3 pivots)
ConstFASPivot5 (best of 5 pivots)
ConstFASPivot10 (best of 10 pivots)
BordaLK
MC4Apx
MC4ApproxLK
MC4Approx/Borda+LS
MC4
MC4LK
MC4+LS
Percentage from optimum
CPU Time (in seconds)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Web Communities



Open questions

- Approximation scheme for partial rank aggregation?
- Does the model accurately capture "good" combined rankings?
  - Back to metasearch?

Open questions

- Hope for other linear ordering problems?
  - Recent results seem to say no:
    - Guruswami, Manokaran, Raghavendra (FOCS 2008): can't do better than ½ for Max Acyclic Subgraph if Unique Games has no polytime algorithms.
    - Bansal, Khot (FOCS 2009): can't do better than 2 for single machine scheduling with precedence to minimize weighted completion time if variant of Unique Games has no polytime algorithms.
    - Svensson (STOC 2010): can't do better than 2 for scheduling identical parallel machines with precedence constraints to minimize schedule length if variant of Unique Games has no polytime algorithms.
- Perhaps prove that 4/3 is best possible given Unique Games?

Obrigado.

Any questions?

dpw@cs.cornell.edu

www.davidpwilliamson.net/work

Open questions

- Linear ordering polytope has integrality gap of 4/3 for weights from full rank aggregation:

  Min $\Sigma_{i,j}$ x(i,j)w(j,i) + x(j,i)w(i,j)

  s.t. x(i,j) + x(j,i) = 1 for all i,j

  x(i,k) + x(k,j) + x(j,i) $\geq$ 1 for all distinct i,j,k

  x(i,j) ¸ 0

when w(i,j) + w(j,i) = 1,

w(i,j) + w(k,j) + w(j,i) ¸ 1.

Is this the worst case for these instances?

Remainder of talk

Approximation algorithms for rank aggregation

- ✓ A very simple 2-approximation algorithm for full rank aggregation
- ✓ Pivoting algorithms
- ✓ A simple, deterministic 2-approximation algorithm for triangle inequality
- ✓ A 1.6-approximation algorithm for full rank aggregation
- ❑ LP-based pivoting

Further results

- To get results for other classes of weights (e.g. for tournaments) and stronger results for rank aggregation, we need linear programming based algorithms.
- Ailon, Charikar, Newman (STOC ’05) and Ailon (SODA ’07) give randomized rounding algorithms; made deterministic by Van Zuylen, Hegde, Jain, W (SODA ’06) and Van Zuylen, W ’07.

Why LP based?

Consider tournaments

$$w(i,j) = \begin{cases} 1 & \text{if } (i,j) \text{ in tournament} \\ 0 & \text{otherwise} \end{cases}$$

$\Rightarrow$ $w_{ij} \equiv 0$

$\Rightarrow$ $\sum_{ij} w_{ij} = 0$

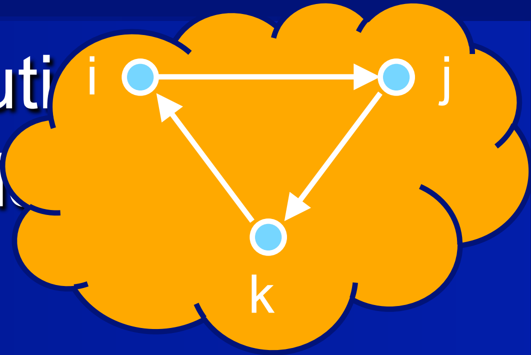
$\Rightarrow$ Lower bound of 0!

$\Rightarrow$ Need better lower bound!

LP based algorithms

Solve LP relaxation, and round soluti

$x(i,j) = 1$ if i before j, 0 otherwi



Min $\Sigma_{i,j}\ x(i,j)w(j,i) + x(j,i)w(i,j)$

s.t. $x(i,j) + x(j,i) = 1$ for all i,j

$x(i,k) + x(k,j) + x(j,i) \geq 1$ for all distinct i,j,k

$x(i,j) \in$ ~~{0,1}~~ ≥ 0

LP based algorithms

Two types of rounding:

1. - Form tournament G=(V,A) that has (i,j) in A if $x(i,j) \geq {}^1/_2$
   - Pivot to get an acyclic solution (where a pivot is chosen similar to before)

2. - Choose a vertex j as pivot

     order i left of j with proba

     order i right of j with prob

   - Recurse on left and right

use method of conditional expectation to derandomize

LP based algorithms: approximation guarantees

1. "Deterministic rounding"

probability constraints:	3

2. "Conditional expectation"

probability constraints:	${}^{5}/_{2}$
triangle inequality constraints (partial rank aggregation):	${}^{3}/_{2}$
full rank aggregation:	${}^{4}/_{3}$

Randomized versions due to Ailon et al. and Ailon; deterministic versions by Van Zuylen et al. and Van Zuylen and W.

Remainder of talk

Approximation algorithms for rank aggregation

- ✓ A very simple 2-approximation algorithm for full rank aggregation
- ✓ Pivoting algorithms
- ✓ A simple, deterministic 2-approximation algorithm for triangle inequality
- ✓ A 1.6-approximation algorithm for partial rank aggregation
- ✓ LP-based pivoting

Combining the two 2-approximations

budget for {i,j}

majority tournament has (i,j) if

$\geq w(j,i)$

$w_{ij} = \min\{w(i,j), w(j,i)\}$

$w_{ij} = \max\{w(i,j), w(j,i)\}$

wil show:
total new cost $\leq$
$(1+\alpha)$ total budget
for $\alpha=0.6$

New cost of forward arc:

$$\alpha z_{ij} + (1-\alpha)\, w_{ij}$$

New cost of backward arc:

$$\alpha z_{ij} + (1-\alpha)\, w_{ij}$$

Combining the two 2-approximations

Forward costs:

$$\alpha\, z_{ij} + (1-\alpha)\, w_{ij} \cdot \alpha\, (2w_{ij}) + (1-\alpha)\, w_{ij}$$

$$\leq (1+\alpha)\, w_{ij}$$

Combining the two 2-approximations

Backward costs:

new cost for backward arc = $\alpha\, z_{ij} + (1-\alpha)\, w_{ij}$

"budget" for backward arc = w_{ij}

Lemma: there exists a pivot such that

new cost of backward arcs ≤

$(1+\alpha)$ (budget of backward arcs)

for $\alpha = 0.6$

$\Rightarrow$ the combined algorithm is a 1.6 approximation algorithm

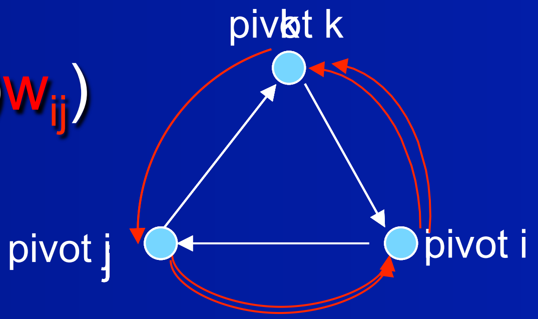
Combining the two 2-approximations – proof of Lemma

Σ_{pivots} (new cost of backward arcs) =

$\Sigma_{\text{directed triangles } t}\, \Sigma_{(i,j) \text{ in } t}\, (\alpha z_{ij} + (1-\alpha) w_{ij})$

Σ_{pivots} (budget of backward arcs) =

$\Sigma_{\text{directed triangles } t}\, \Sigma_{(i,j) \text{ in } t}\, w_{ij}$



Fact: for $\alpha=0.6$

$\Sigma_{(i,j) \text{ in } t}\, (\alpha z_{ij} + (1-\alpha) w_{ij}) \le (1+\alpha)\Sigma_{(i,j) \text{ in } t}\, w_{ij}$ for all directed triangles t

$\Rightarrow$ there exists a pivot such that

new cost of backward arcs $\le$ 1.6 (budget of backward arcs)

Clustering

- Input:
  - Set of elements V
  - Pairwise information $w^+\{i,j\}$, $w^-\{i,j\}$
  - Assumption: weights satisfy
    - triangle inequality or
    - probability constraints
- Goal:
  - Find a clustering that minimizes

$$\Sigma_{i,j \text{ together}} w^-\{i,j\} + \Sigma_{i,j \text{ separated}} w^+\{i,j\}$$

Clustering

“Majority tournament” ⇔

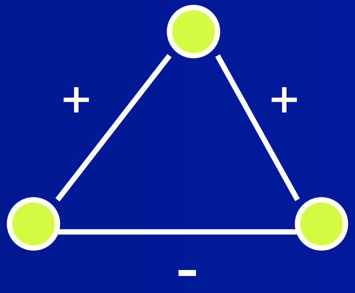
- ‘+’ edge {i,j} if $w^{+}\{i,j\} \geq w^{-}\{i,j\}$
- ‘-’edge {i,j} if $w^{-}\{i,j\} \geq w^{+}\{i,j\}$

Pivoting on vertex k:

- If {i,k} is a ‘+’ edge, put i in same cluster as k
- If {i,k} is a ‘-’ edge, separate i from k

Recurse on vertices separated from k

“Directed triangle” ⇔



More results

- Kenyon-Mathieu and Schudy ’07 give an approximation scheme for full rank aggregation.
- Empirical study of these algorithms in progress (Van Zuylen).