Sorting under partial information (without the ellipsoid algorithm)

Jean Cardinal
ULB

Samuel Fiorini
ULB

Gwenaël Joret
ULB

Raphaël Jungers
UCL/MIT

Ian Munro
Waterloo

Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries

Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries

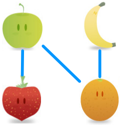
Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries



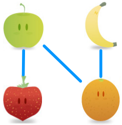
Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \dots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries



 OR  ?

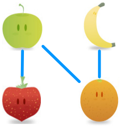
Sorting by comparisons under partial information

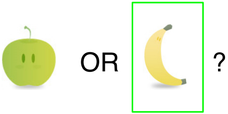
Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries





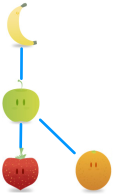
Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \dots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: $\#$queries



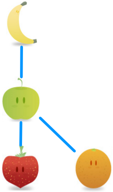
Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries



 OR  ?

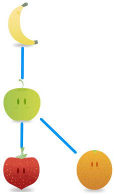
Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \dots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries



OR

?

Sorting by comparisons under partial information

Input:

- set $V = \{v_1, \ldots, v_n\}$, with unknown linear order $\leqslant$
- poset $P = (V, \leqslant_P)$ compatible with $(V, \leqslant)$

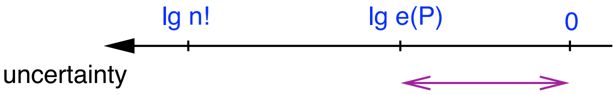
Goal: Discover $\leqslant$ by making queries "is $v_i \leqslant v_j$?"

Objective function: #queries

Lower bound on #queries

Every algorithm can be forced to a #queries that is *at least*

$$\boxed{\text{lg(\#linear extensions of P)} = \lg e(P)}$$



Results

- **Known results**

	#queries	complexity
Fredman 1976	$\lg e(P) + 2n$	super-polynomial
Kahn & Saks 1984	$O(\lg e(P))$	super-polynomial
Kahn & Kim 1995	$O(\lg e(P))$	polynomial (ellipsoid alg.)

Results

- **Known results**

	#queries	complexity
Fredman 1976	$\lg e(P) + 2n$	super-polynomial
Kahn & Saks 1984	$O(\lg e(P))$	super-polynomial
Kahn & Kim 1995	$O(\lg e(P))$	polynomial (ellipsoid alg.)

- **Our contribution**: two ellipsoid-free algorithms

	#queries		complexity
Algorithm 1	$(1+\varepsilon)\lg e(P) + O_\varepsilon(n)$	$\forall \varepsilon > 0$	$O(n^{2.5})$
Algorithm 2	$O(\lg e(P))$		$O(n^{2.5})$

+ preprocessing in $O(n^{2.5})$, sort linear in #queries and n
+ better understanding of *entropy* of P

Computing the lower bound

Computing $e(P)$ is #P-complete — Brightwell & Winkler 1991

~~Computing~~ Approximating the lower bound

Computing $e(P)$ is #P-complete Brightwell & Winkler 1991

As Kahn & Kim 1995, use the *entropy* $H(\bar{P})$

~~Computing~~ Approximating the lower bound

Computing $e(P)$ is #P-complete — Brightwell & Winkler 1991

As Kahn & Kim 1995, use the *entropy* $H(\bar{P})$

Why?

~~Computing~~ Approximating the lower bound

Computing $e(P)$ is #P-complete — Brightwell & Winkler 1991

As Kahn & Kim 1995, use the *entropy* $H(\bar{P})$

Why?

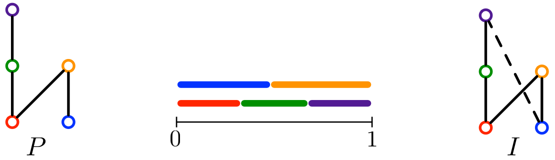
- $\boxed{\lg e(P) = \Theta(nH(\bar{P}))}$ — Kahn & Kim 1995
- $H(\bar{P})$ can be "computed" in poly-time using the ellipsoid algorithm (or an interior point algorithm)

Entropy

"New" definition – original definition due to Körner 1973, applies to any graph

$\{(y_{v^-}, y_{v^+})\}_{v \in V}$ is *consistent* with P if

- $\forall v \in V$: (y_{v^-}, y_{v^+}) open interval $\subseteq (0,1)$
- $v \leqslant_P w \implies y_{v^+} \leq y_{w^-}$



$$H(P) := \min \left\{ -\frac{1}{n} \sum_{v \in V} \lg x_v \;\middle|\; \exists \{(y_{v^-}, y_{v^+})\}_{v \in V} \text{ consistent with } P \text{ s.t. } x_v = y_{v^+} - y_{v^-} \quad \forall v \in V \right\}$$

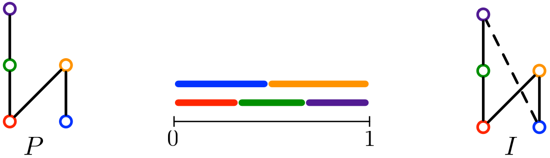
$$H(P) = \frac{1}{n} \times \text{“information” in } P$$

Entropy

"New" definition – original definition due to Körner 1973, applies to any graph

$\{(y_{v^-}, y_{v^+})\}_{v\in V}$ is *consistent* with P if

- $\forall v \in V$: (y_{v^-}, y_{v^+}) open interval $\subseteq (0,1)$
- $v \leqslant_P w \implies y_{v^+} \leq y_{w^-}$



$$H(P) := \min \Big\{ -\frac{1}{n}\sum_{v\in V} \lg x_v \mid \exists\{(y_{v^-}, y_{v^+})\}_{v\in V} \text{ consistent with } P \text{ s.t. } x_v = y_{v^+} - y_{v^-} \quad \forall v \in V \Big\}$$

$$\boxed{H(\bar{P}) := \lg n - H(P)} = \frac{1}{n} \times \text{"uncertainty" in } P$$

Bounds I

"Additive" bound via the order polytope

$$O(P) := \{y \in [0,1]^V \mid v \leqslant_P w \Longrightarrow y_v \leq y_w\}$$

Easy fact 1. volume $O(P) = \dfrac{e(P)}{n!}$

Easy fact 2.

$$\{(y_{v^-}, y_{v^+})\}_{v \in V} \text{ consistent with } P$$

$$\Longrightarrow \prod_{v \in V} [y_{v^-}, y_{v^+}] \subseteq O(P)$$

$$\Longrightarrow \prod_{v \in V} x_v \leq \text{volume } O(P)$$

Corollary.

$$nH(\bar{P}) \leq \lg e(P) + n \lg e$$

K&K 1995

Bounds II

"multiplicative" bound

$$\lg e(P) \leq nH(\bar{P}) \leq c \cdot \lg e(P)$$ K&K 1995

where $c = 1 + 7 \lg e \simeq 11.1$

Bounds II

"multiplicative" bound

$$\lg e(P) \leq nH(\bar{P}) \leq c \cdot \lg e(P)$$ K&K 1995

where $c = 1 + 7 \lg e \simeq 11.1$

Rmk: Lower bound tight, but not upper bound

- K&K conjectured

$$nH(\bar{P}) \leq (1 + \lg e) \cdot \lg e(P)$$ $(1 + \lg e \simeq 2.44)$

Bounds II

"multiplicative" bound

$$\lg e(P) \leq nH(\bar{P}) \leq c \cdot \lg e(P) \qquad \text{K\&K 1995}$$

where $c = 1 + 7 \lg e \simeq 11.1$

Rmk: Lower bound tight, but not upper bound

- K&K conjectured

$$nH(\bar{P}) \leq (1 + \lg e) \cdot \lg e(P) \qquad (1 + \lg e \simeq 2.44)$$

- We prove

$$nH(\bar{P}) \leq 2 \cdot \lg e(P)$$

(tight)

K&K's algorithm

Key lemma:

$\exists$ incomparable pair a, b s.t.

$$\max\left\{nH(\bar{P}(a < b)), nH(\bar{P}(a > b))\right\} \leq nH(\bar{P}) - c,$$

where $c \simeq 0.2$

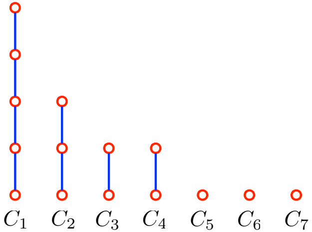
Algorithm:

1. Repeat:
   1.1 Compute $H(P)$ and optimal solution x^*
   1.2 Find good incomparable pair a, b using x^*
   1.3 Compare a and b
   1.4 Update P

$\#\text{steps} = O(nH(\bar{P})) = O(\lg e(P))$

Greedy

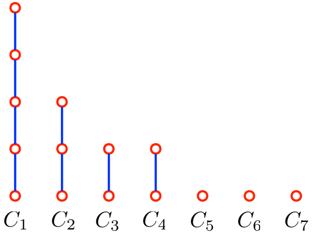
Greedy chain decomposition of P $\quad\rightarrow\quad$ $U := C_1 \cup \cdots \cup C_k$



$$H(\bar{U}) = \sum_{i=1}^{k} -\frac{|C_i|}{n} \lg \frac{|C_i|}{n}$$

Greedy

Greedy chain decomposition of $P \quad \rightarrow \quad U := C_1 \cup \cdots \cup C_k$



$$H(\bar{U}) = \sum_{i=1}^{k} -\frac{|C_i|}{n} \lg \frac{|C_i|}{n}$$

From perfectness of incomparability graph of P:

$$\boxed{H(\bar{U}) \leq (1+\varepsilon)H(\bar{P}) + O_\varepsilon(1) \qquad \forall \varepsilon > 0}$$

CFJJM 2009

Extends to every perfect graph!

Algorithms

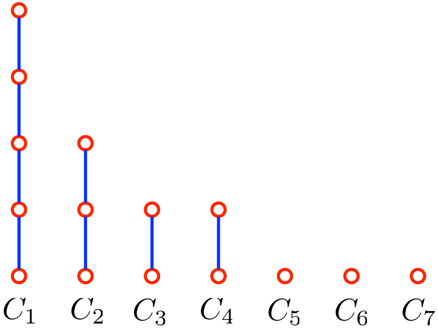
	#queries	complexity
Algorithm 1	$(1+\varepsilon)\lg e(P) + O_\varepsilon(n) \quad \forall \varepsilon > 0$	$O(n^{2.5})$
Algorithm 2	$O(\lg e(P))$	$O(n^{2.5})$

Algorithm 1: greedy + merge sort

Algorithm 2: greedy + "cautious" merge sort

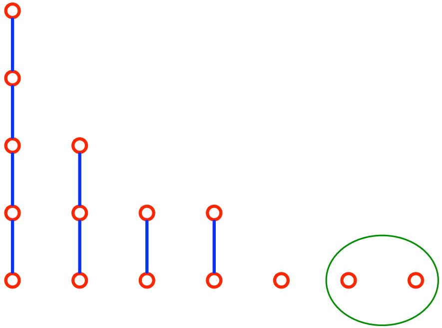
Algorithm 1

1. Compute greedy chain decomposition of P
2. Iteratively merge two smallest chains



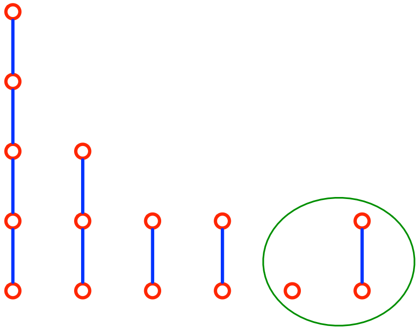
Algorithm 1

1. Compute greedy chain decomposition of P
2. Iteratively merge two smallest chains



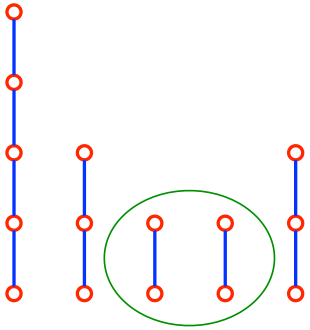
Algorithm 1

1. Compute greedy chain decomposition of P
2. Iteratively merge two smallest chains



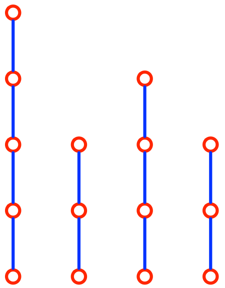
Algorithm 1

1. Compute greedy chain decomposition of P
2. Iteratively merge two smallest chains



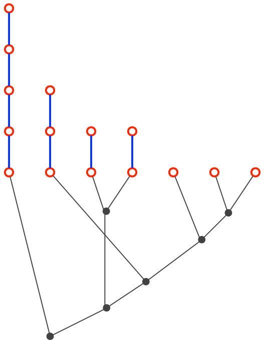
Algorithm 1

1. Compute greedy chain decomposition of P
2. Iteratively merge two smallest chains



ETC.

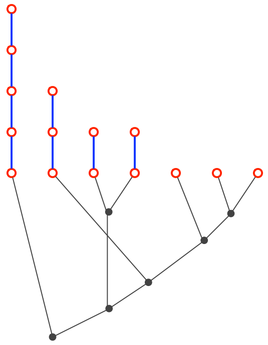
Algorithm 1



Huffman trees: average root-to-leaf distance in tree at most

$$\left(\sum_{i=1}^{k} -\frac{|C_i|}{n} \lg \frac{|C_i|}{n} \right) + 1 = H(\bar{U}) + 1$$

Algorithm 1



Huffman trees: average root-to-leaf distance in tree at most

$$\left(\sum_{i=1}^{k} -\frac{|C_i|}{n} \lg \frac{|C_i|}{n}\right) + 1 = H(\bar{U}) + 1$$

$\Rightarrow \ldots \Rightarrow$ at most $(H(\bar{U}) + 1)n$ comparisons

Algorithm 1

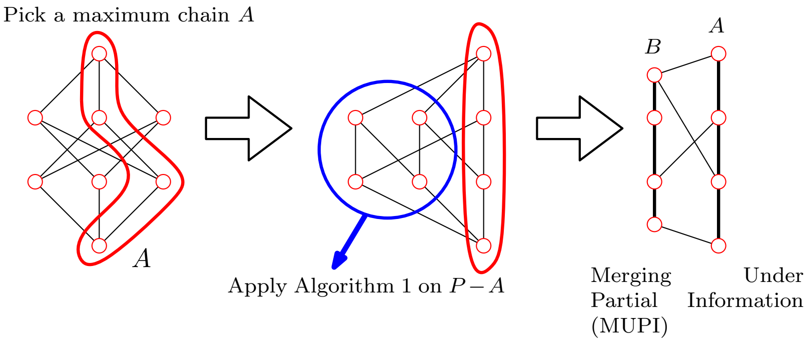
Huffman trees: average root-to-leaf distance in tree at most

$$\left(\sum_{i=1}^{k} -\frac{|C_i|}{n} \lg \frac{|C_i|}{n}\right) + 1 = H(\bar{U}) + 1$$

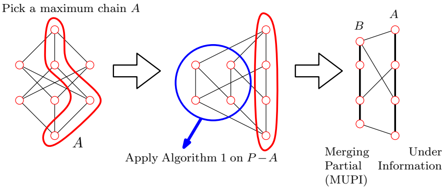
$\Rightarrow \ldots \Rightarrow$ at most $(H(\bar{U}) + 1)n$ comparisons

$$\begin{aligned} (H(\bar{U}) + 1)n &\leq (1+\varepsilon)nH(\bar{P}) + O_\varepsilon(n) && \text{greedy} \\ &\leq (1+\varepsilon)\big(\lg e(P) + n \lg e\big) + O_\varepsilon(n) && \text{K\&K's additive bd} \\ &= (1+\varepsilon)\lg e(P) + O_\varepsilon(n) \end{aligned}$$

Algorithm 2

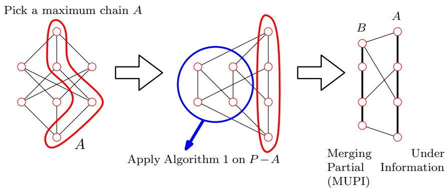

Pick a maximum chain A
A
Apply Algorithm 1 on $P - A$
B
A
Merging Under Partial Information (MUPI)

Algorithm 2



#comparisons in step 2 at most

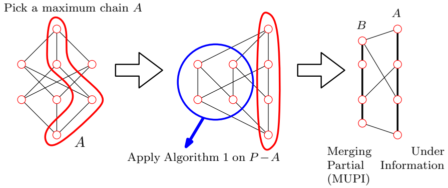
Algorithm 2



#comparisons in step 2 at most

$$(1+\varepsilon)\lg e(P-A) + O_{\varepsilon}(|P-A|)$$

Algorithm 2



#comparisons in step 2 at most

$$(1+\varepsilon)\lg e(P-A) + O_\varepsilon(|P-A|)$$

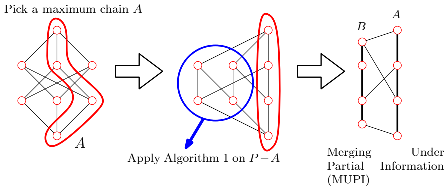
[Interlude] An easy lemma (take all intervals of length $x_v = \frac{1}{|A|}$):

$$H(\bar{P}) \geq -\lg \frac{|A|}{n}$$

$\Rightarrow |A| \geq 2^{-H(\bar{P})} n$

$\Rightarrow |P-A| \leq n\left(1 - 2^{-H(\bar{P})}\right) \leq \ln 2 \cdot nH(\bar{P})$ (using $1 - 2^{-x} \leq \ln 2 \cdot x$)

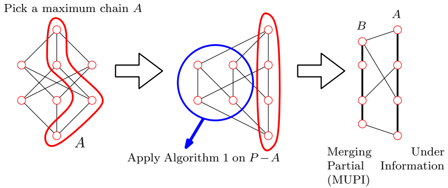
Algorithm 2



#comparisons in step 2 at most

$$
\begin{aligned}
&(1+\varepsilon)\lg e(P-A) + O_\varepsilon(|P-A|) \\
\leq\, &(1+\varepsilon)\lg e(P-A) + O_\varepsilon\left(\ln 2 \cdot nH(\bar{P})\right) \\
\leq\, &(1+\varepsilon)\lg e(P) + O_\varepsilon\left(\lg e(P)\right) \qquad \text{K\&K's multiplicative bd} \\
=\, &O_\varepsilon\left(\lg e(P)\right)
\end{aligned}
$$

Algorithm 2



#comparisons in step 2 at most

$$
\begin{aligned}
& (1+\varepsilon) \lg e(P-A) + O_\varepsilon(|P-A|) \\
\le & (1+\varepsilon) \lg e(P-A) + O_\varepsilon\left(\ln 2 \cdot nH(\bar{P})\right) \\
\le & (1+\varepsilon) \lg e(P) + O_\varepsilon\left(\lg e(P)\right) \qquad \text{K\&K's multiplicative bd} \\
= & O_\varepsilon\left(\lg e(P)\right)
\end{aligned}
$$

What about partial information P' in step 3?

$P' \supseteq P \Rightarrow \lg e(P') \le \lg e(P)$

$\Rightarrow$ enough to solve MUPI = Merging under Partial Information!


MUPI

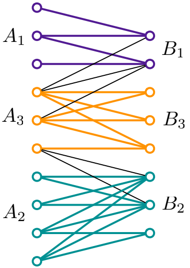
Merging under partial information

Overview for MUPI:

1. Compute entropy exactly (easier) Körner and Marton 1988
2. Use Hwang-Lin merging algorithm guided by x^*
3. Update x^*

Posets of width ≤ 2

In that special case, the incomparability graph of P is bipartite

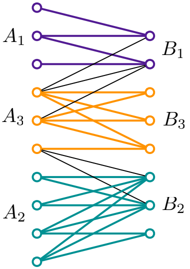


Körner and Marton 1988:

- optimal solution for entropy has "block structure"
- can be computed via a greedy algorithm

Posets of width ≤ 2

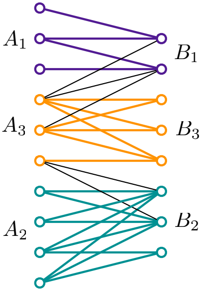
bipartite incomparability graphs $\implies$ x^* defining $H(P)$ has an even nicer structure



- A_i interval of A, B_i interval of B, same ordering
- $x^*_v = (|A_i| + |B_i|)/n|A_i|$ whenever $v \in A_i$

Can compute $H(\bar{P})$ and x^* in time $O(n^2 \log^2 n)$

Solving MUPI - general ideas



Compute entropy and x^*

Apply Hwang-Ling merging algorithm on each component $A_i \cup B_i$ with $|A_i| \geq |B_i|$, in a certain order

Update x^* locally after each merging (details omitted)

Overall #comparisons is $\leq 3nH(\bar{P})$

Thank You!