ALL PAIRS SHORTEST PATHS
in
UNDIRECTED GRAPHS
with
INTEGRAL WEIGHTS

Avi Shoshan

Uri Zwick

Tel Aviv University

THE APSP PROBLEM

Given

a weighted **undirected** graph on n vertices
with weights taken from the set

$$\{0, 1, \ldots, M\}.$$

Find

SHORTEST PATHS

between all pairs of vertices in the graph.

Directed graphs with real weights

Dijkstra '59 Johnson '77 Fredman, Tarjan '87	$nm + n^2 \log n$

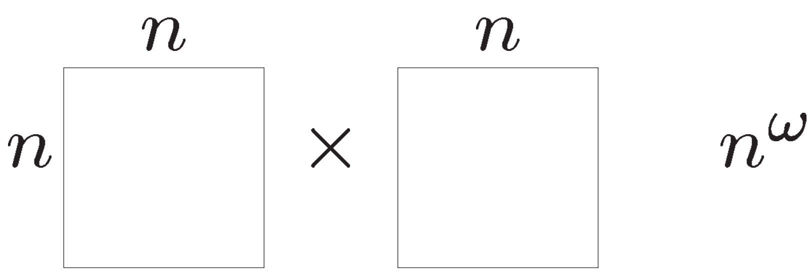
Directed graphs with small integer weights

Alon, Galil Margalit '91	$n^{\frac{3+\omega}{2}} \le n^{2.688}$
Zwick '98	$n^{2+\frac{1-\alpha(\omega-1)}{4-\omega-2\alpha}} \le n^{2.575}$

Undirected graphs with small integer weights

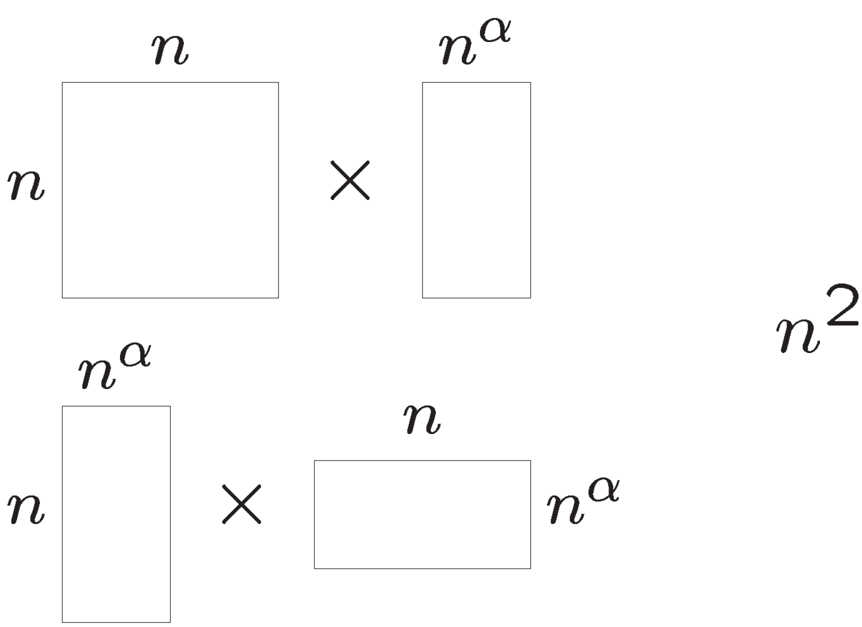
Seidel '92	$n^{\omega} \le n^{2.376}$
Galil, Margalit '92	$M^{\frac{\omega+1}{2}} \cdot n^{\omega}$
Shoshan, Zwick '99	$M \cdot n^{\omega}$

SQUARE MATRIX MULTIPLICATION



$\omega < 2.376$ [Coppersmith, Winograd '90]

RECTANGULAR MATRIX MULTIPLICATION



$\alpha > 0.294$ [Coppersmith '97]

OVER ANY **RING**

APPROXIMATION ALGORITHMS

Unweighted, undirected graphs

			error
Aingworth, Chekuri Indyk, Motwani	'96	$n^{5/2}$	$+2$
Dor, Halperin Zwick	'96	$n^{7/3}$	$+2$

Weighted, undirected graphs

			error
Cohen Zwick	'97	$n^{3/2}m^{1/2}$	$\times 2$
		n^2	$\times 3$

Weighted, directed graphs

		error
Zwick '98	$\frac{n^\omega}{\epsilon} \cdot \log M$	$\times (1+\epsilon)$

DISTANCE PRODUCTS

$$\begin{bmatrix} -6 & -3 & -10 \\ 2 & 5 & -2 \\ -1 & -7 & -5 \end{bmatrix} = \begin{bmatrix} 1 & -3 & 7 \\ +\infty & 5 & +\infty \\ 8 & 2 & -5 \end{bmatrix} \star \begin{bmatrix} 8 & +\infty & -4 \\ -3 & 0 & -7 \\ 5 & -2 & 1 \end{bmatrix}$$

$$C \quad = \quad A \quad \star \quad B$$

$$c_{ij} = \min_k \{ a_{ik} + b_{kj} \}$$

DISTANCES IN GRAPHS

$$D = \begin{bmatrix} 0 & & & 5 & -8 \\ & 0 & & & +\infty \\ & & \ddots & & \\ +\infty & & & 0 & \\ 7 & -5 & & & 0 \end{bmatrix}$$

If D contains the edge weights of a graph then D^n contains all the distances.

CAN WE USE

FAST MATRIX MULTIPLICATION

ALGORITHMS

DIRECTLY

TO COMPUTE

DISTANCE PRODUCTS ?

NO !

$(\mathbb{R}, \min, +)$

IS NOT A RING

min has no inverse

COMPUTING DISTANCE PRODUCTS

The distance product of two $n \times n$ matrices with elements from

$$\{-L, \ldots, 0, \ldots, L\} \cup \{+\infty\}$$

can be computed in

$$L \cdot n^{\omega} \text{ time.}$$

(Yuval '76)
(Alon, Galil, Margalit '91)

Main idea:

To compute the distance product $A \star B$, compute the algebraic product $(x^A)(x^B)$, where x is an indeterminant.

Open problem:

Find a better algorithm !

Using matrix multiplication to compute distance products

$$\begin{bmatrix} c_{11} & c_{12} & \\ c_{21} & c_{22} & \\ & & \ddots \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \\ a_{21} & a_{22} & \\ & & \ddots \end{bmatrix} \star \begin{bmatrix} b_{11} & b_{12} & \\ b_{21} & b_{22} & \\ & & \ddots \end{bmatrix}$$

$$c_{ij} = \min_k \{ a_{ik} + b_{kj} \}$$

$$0 \le a_{ij}, b_{ij} \le L$$

$$C' = \begin{bmatrix} x^{a_{11}} & x^{a_{12}} & \\ x^{a_{21}} & x^{a_{22}} & \\ & & \ddots \end{bmatrix} \star \begin{bmatrix} x^{b_{11}} & x^{b_{12}} & \\ x^{b_{21}} & x^{b_{22}} & \\ & & \ddots \end{bmatrix}$$

x - an **indeterminant**

Compute the normal algebraic product of the two polynomial matrices

$$c'_{ij} = \sum_k x^{a_{ik}+b_{kj}}$$

$$c_{ij} = \text{deg}^*(c'_{ij})$$

$$n^{\omega}$$

polynomial products

$$\times$$

$$L \cdot \log n$$

time per each
polynomial product

$$\Downarrow$$

The distance products of two $n \times n$ matrices with elements taken from

$$\{-L, \ldots, 0, \ldots, L\} \cup \{+\infty\}$$

can be computed in about

$L \cdot n^{\omega}$ time

REPEATED POWERING

Suppose all weights are 1.

All entries in D are from $\{1, +\infty\}$

$F \leftarrow D$
repeat $\lceil \log_2 n \rceil$ **times**
$F \leftarrow F \star F$

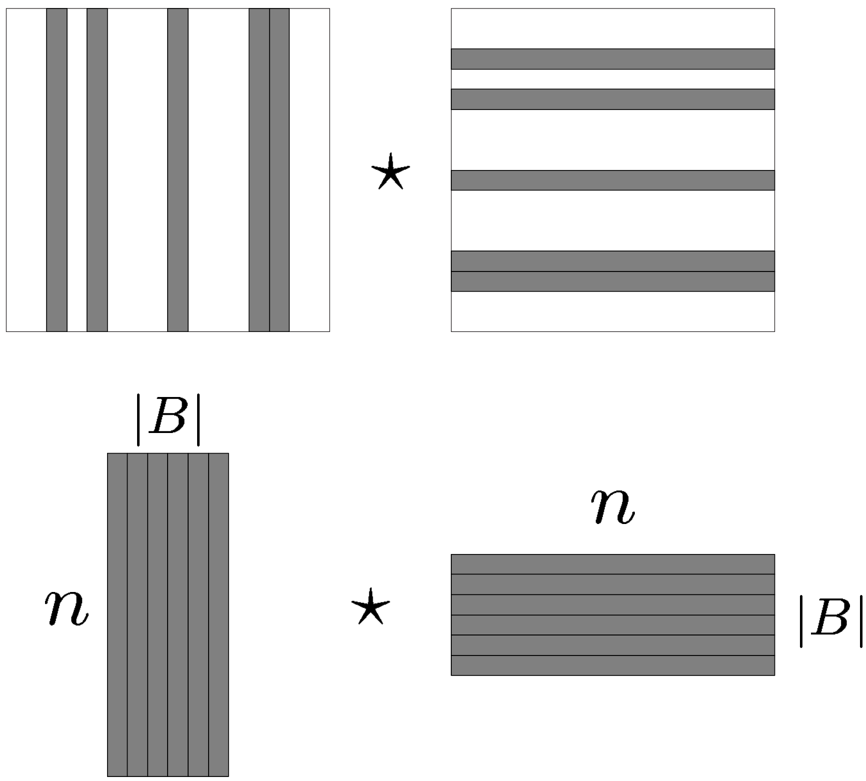
Naive implementation takes

$n^3 \log n$ time

Using the $L \cdot n^\omega$ algorithm

we get $n \cdot n^\omega = n^{\omega+1} \gg n^3$ time

REPEATED POWERING with SAMPLING



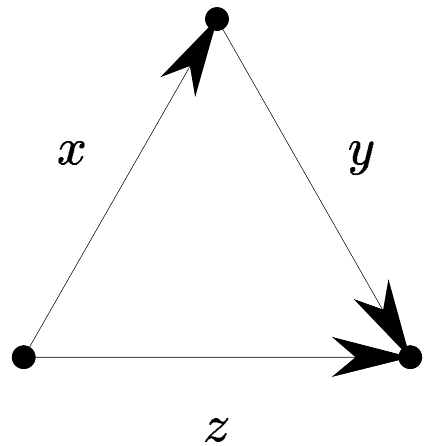
In each iteration, Choose a subset B of the vertices and replace the product

$$F \star F$$

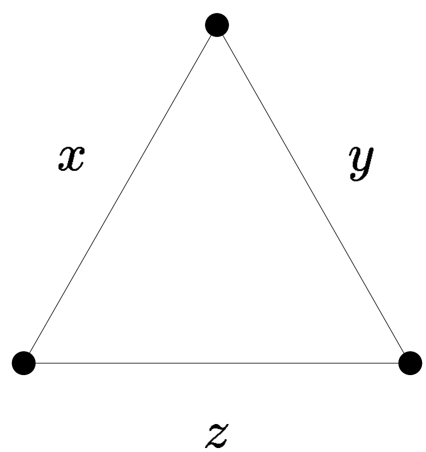
by

$$F[*, B] \star F[B, *]$$

Undirected Graphs
vs.
Directed Graphs



$$z \leq x + y$$



$$|x - y| \leq z \leq x + y$$

SEIDEL'S ALGORITHM

```
algorithm SEIDEL(A)

if A = J then
    return J − I
else
    C ← SEIDEL(A²)
    X ← CA , deg ← Ae − 1
    d_ij ← 2c_ij − [x_ij < c_ij deg_j]
    return D
end
```

$$\textbf{algorithm}\ \ \mathrm{SEIDEL}(A)$$

$$\textbf{if}\ A = J\ \textbf{then}$$

$$\quad \textbf{return}\ J - I$$

$$\textbf{else}$$

$$\quad C \leftarrow \mathrm{SEIDEL}(A^2)$$

$$\quad X \leftarrow CA\,,\ \mathrm{deg} \leftarrow Ae - 1$$

$$\quad d_{ij} \leftarrow 2c_{ij} - [x_{ij} < c_{ij}\,\mathrm{deg}_j]$$

$$\quad \textbf{return}\ D$$

$$\textbf{end}$$

SIMPLE!

Uses both **Boolean** and **integer** products

Works in **BINARY**

Does **not** work on weighted graphs

THE ALGORITHM OF GALIL and MARGALIT

COMPLICATED!

Uses only BOOLEAN matrix products

Works in TRENARY

Works on weighted graphs!

Running time: $M^{\frac{\omega+1}{2}} \cdot n^{\omega}$

THE NEW ALGORITHM

(Relatively) SIMPLE

Uses DISTANCE PRODUCTS

Works in BINARY

Works on weighted graphs!

Running time: $M \cdot n^{\omega}$

NEW ALGORITHM for UNWEIGHTED GRAPHS

algorithm **USP**(A)

1. $\ell \leftarrow \lceil \log_2 n \rceil$

   $A_0 \leftarrow I \,,\; B_0 \leftarrow A \vee I$

   for $k \leftarrow 1$ **to** ℓ **do**

   $\quad A_k \leftarrow A_{k-1} \cdot B_{k-1}$

   $\quad B_k \leftarrow B_{k-1} \cdot B_{k-1}$

   end

2. $C_\ell \leftarrow 1 \;;\; D_\ell \leftarrow 1$

   $P_\ell \leftarrow I \;;\; Q_\ell \leftarrow 0$

   for $k \leftarrow \ell - 1$ **downto** 0

   $$C_k \leftarrow [(P_{k+1} \cdot A_k) \wedge \;\; C_{k+1}] \;\vee\; [(Q_{k+1} \cdot A_k) \wedge \neg C_{k+1}]$$

   $$D_k \leftarrow [(P_{k+1} \cdot B_k) \wedge \;\; C_{k+1}] \;\vee\; [(Q_{k+1} \cdot B_k) \wedge \neg C_{k+1}]$$

   $\quad P_k \leftarrow P_{k+1} \vee Q_{k+1}$

   $\quad Q_k \leftarrow D_k \wedge \neg C_k$

   end

3. $\Delta \leftarrow \sum_{k=0}^{\ell} 2^k (\neg C_k)$

algorithm **WUSP**(D)

1\. $\ell \leftarrow \lceil \log_2 n \rceil$, $m \leftarrow \log_2 M$

for $k \leftarrow 1$ **to** $m+1$ **do**

$D \leftarrow \mathit{clip}(D \star D, 0, 2M)$

2\. $A_0 \leftarrow D - M$

for $k \leftarrow 1$ **to** ℓ **do**

$A_k \leftarrow \mathit{clip}(A_{k-1} \star A_{k-1}, -M, M)$

3\. $C_\ell \leftarrow -M$

$P_\ell \leftarrow \mathit{clip}(D, 0, M)$

$Q_\ell \leftarrow +\infty$

for $k \leftarrow \ell - 1$ **downto** 0 **do**

$C_k \leftarrow$

$[\mathit{clip}(P_{k+1} \star A_k, -M, M) \wedge C_{k+1}] \vee$

$[\mathit{clip}(Q_{k+1} \star A_k, -M, M) \bar{\wedge} C_{k+1}]$

$P_k \leftarrow P_{k+1} \vee Q_{k+1}$

$Q_k \leftarrow \mathit{chop}(C_k, 1 - M, M)$

end

4\. **for** $k \leftarrow 1$ **to** ℓ **do**

$B_k \leftarrow (C_k \geq 0)$

$B_0 \leftarrow (0 \leq P_0 < M)$

$R \leftarrow P_0 \bmod M$

$\Delta \leftarrow M \cdot \sum_{k=0}^{\ell} 2^k B_k + R$

BASIC RELATIONS

$$\begin{aligned} \delta &= \{\delta(i,j)\}_{ij} && \text{(distances)} \\ \delta_X &= \{\delta(i,j) \in X\}_{ij} && \text{(bits)} \end{aligned}$$

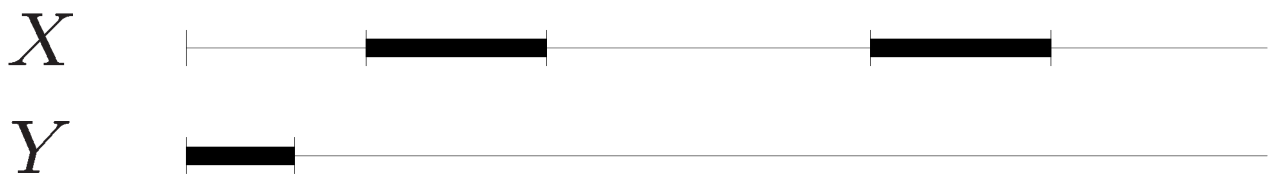
$$\begin{aligned} \delta_X \vee \delta_Y &= \delta_{X \cup Y} \\ \delta_X \wedge \delta_Y &= \delta_{X \cap Y} \\ \neg \delta_X &= \delta_{X^c} \end{aligned}$$

$$\delta_{X+Y} \leq \delta_X \cdot \delta_Y \leq \delta_{X \circ Y}$$

$$\begin{aligned} X + Y &= \{x + y \,|\, x \in X \,,\; y \in Y\} \\ X \circ Y &= \bigcup_{x \in X \,,\, y \in Y} [\,|x - y|\,,\, x + y\,] \end{aligned}$$

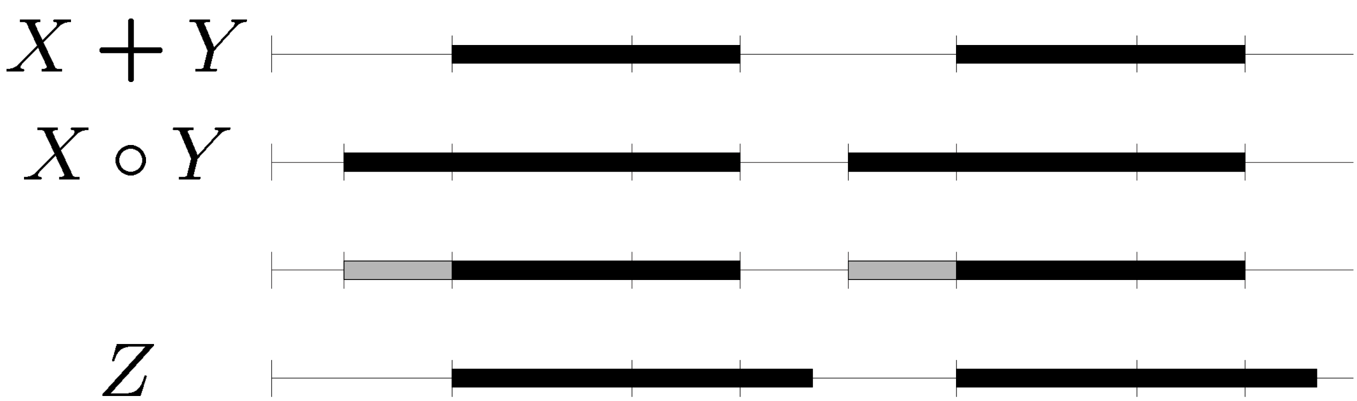
GRAPHICAL DESCRIPTION

$$\delta_{X+Y} \ \leq \ \delta_X \cdot \delta_Y \ \leq \ \delta_{X \circ Y}$$



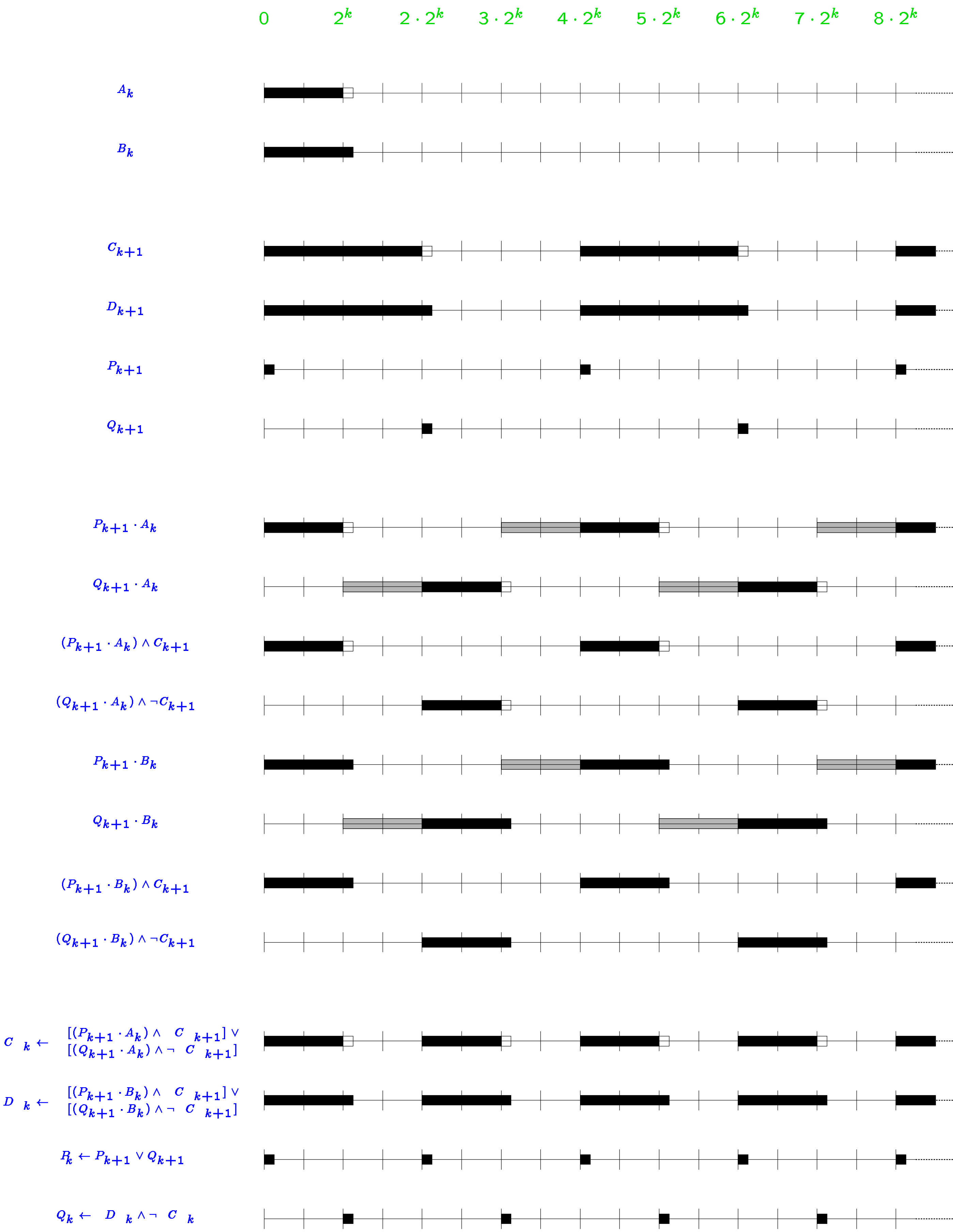
$$X + Y \ = \ \{x + y \mid x \in X \ , \ y \in Y\}$$

$$X \circ Y \ = \bigcup_{x \in X \ , \, y \in Y} [|x - y|, x + y]$$



$$(X \circ Y) \cap Z \ = \ X + Y$$

$$\Rightarrow \ (\delta_X \cdot \delta_Y) \wedge \delta_Z \ = \ \delta_{X+Y}$$

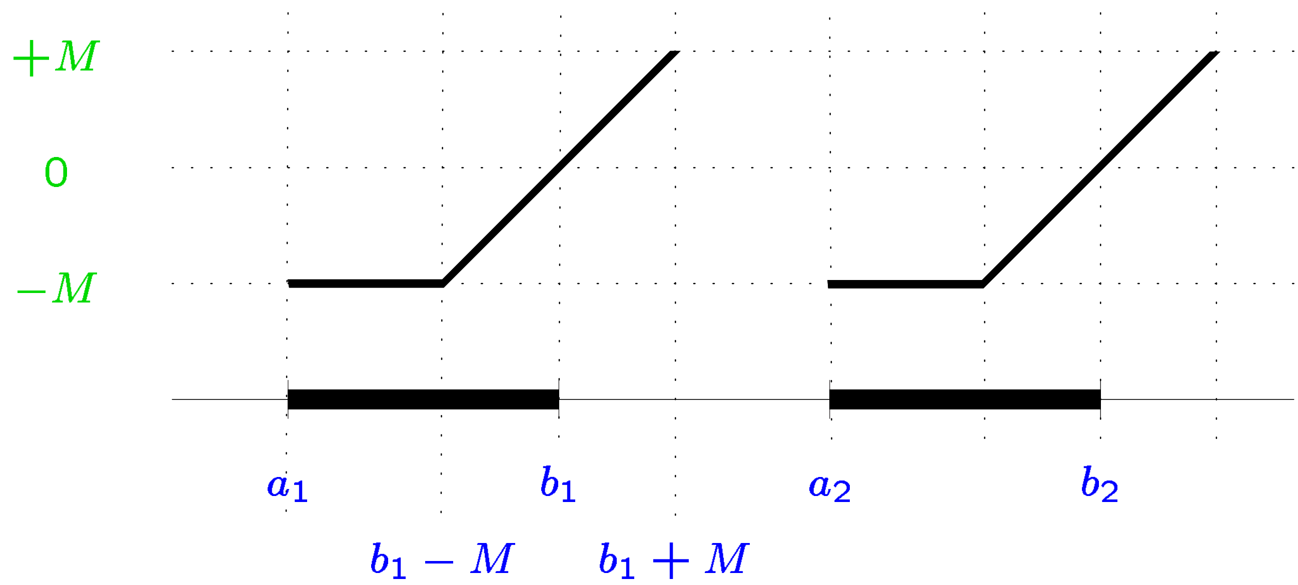

0
2^k
$2 \cdot 2^k$
$3 \cdot 2^k$
$4 \cdot 2^k$
$5 \cdot 2^k$
$6 \cdot 2^k$
$7 \cdot 2^k$
$8 \cdot 2^k$
A_k
B_k
C_{k+1}
D_{k+1}
P_{k+1}
Q_{k+1}
$P_{k+1} \cdot A_k$
$Q_{k+1} \cdot A_k$
$(P_{k+1} \cdot A_k) \wedge C_{k+1}$
$(Q_{k+1} \cdot A_k) \wedge \neg C_{k+1}$
$P_{k+1} \cdot B_k$
$Q_{k+1} \cdot B_k$
$(P_{k+1} \cdot B_k) \wedge C_{k+1}$
$(Q_{k+1} \cdot B_k) \wedge \neg C_{k+1}$
$C_k \leftarrow [(P_{k+1} \cdot A_k) \wedge C_{k+1}] \vee [(Q_{k+1} \cdot A_k) \wedge \neg C_{k+1}]$
$D_k \leftarrow [(P_{k+1} \cdot B_k) \wedge C_{k+1}] \vee [(Q_{k+1} \cdot B_k) \wedge \neg C_{k+1}]$
$P_k \leftarrow P_{k+1} \vee Q_{k+1}$
$Q_k \leftarrow D_k \wedge \neg C_k$

THE WEIGHTED CASE

$$\delta \;=\; \{\delta(i,j)\}_{ij} \quad \text{(distances)}$$

$$X \;=\; \bigcup_r \, [a_r, b_r]$$

$$(\Delta_X)_{ij} \;=\; \begin{cases} -M & \text{if } \;\delta(i,j) \in [a_r, b_r - M] \\ \delta(i,j) - b_r & \text{if } \;\delta(i,j) \in [b_r - M, b_r + M] \\ +\infty & \text{otherwise} \end{cases}$$



$$\Delta_{X \bullet Y} \;\leq\; \Delta_X \star \Delta_Y \;\leq\; \Delta_{X+Y}$$

clipped to

$[-M, M]$

CONCLUDING REMARKS and OPEN PROBLEMS

Undirected graphs

Up to a lagarithmic factor, the APSP problem on graphs with edge weights taken from $\{0, 1, \ldots, M\}$ is equivalent to the distance product problem with elements taken from the same range.

To get an improved algorithm for the APSP problem we must get an improved algorithm for computing distance products.

CONCLUDING REMARKS and OPEN PROBLEMS

Directed graphs

Is the directed case really harder than the undirected case?

Is there a faster algorithm for the APSP problem for directed case? (Best bound is currently $n^{2.575}$)

Distance products

Is there a faster algorithm for computing distance products? (Best bound is currently $L \cdot n^{\omega}$)

Is there an $n^{3-\epsilon} \cdot \log L$ time algorithm for computing distance products?