The Knuth-Yao Quadrangle Inequality Speedup is a Consequence of Total Monotonicity

Wolfgang W. Bein	(University of Nevada)
Mordecai J. Golin	(Hong Kong UST)
Lawrence L. Larmore	(University of Nevada)
Yan Zhang	(Hong Kong UST)

Motivation

- Nothing new: material here goes back 20-30 years.
- There are two classic Dynamic Programming Speedups in the literature
  - Knuth-Yao Quadrangle Inequality Speedup
  - SMAWK Algorithm for Totally Monotone Matrices
- They “feel” similar. Are they related?
- Both techniques have been used quite often in improving DP algorithms for various type of constrained source coding.

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Background

Background

- Kunth-Yao Quadrangle Inequality Speedup
  - D. E. Knuth (1971) and F. F. Yao (1980,1982)
  - $\Theta(n)$ speedup: $O(n^3)$ down to $O(n^2)$

Background

- Kunth-Yao Quadrangle Inequality Speedup
  - D. E. Knuth (1971) and F. F. Yao (1980,1982)
  - $\Theta(n)$ speedup: $O(n^3)$ down to $O(n^2)$
- SMAWK Algorithm for finding
  Row Minima of Totally Monotone Matrices
  - A. Aggarwal, M. M. Klawe, S. Moran, P. Shor, R. Wilber (1986)
  - $\Theta(n)$ speedup: $O(n^2)$ down to $O(n)$

Background

- Kunth-Yao Quadrangle Inequality Speedup
  - D. E. Knuth (1971) and F. F. Yao (1980,1982)
  - $\Theta(n)$ speedup: $O(n^3)$ down to $O(n^2)$
- SMAWK Algorithm for finding
  Row Minima of Totally Monotone Matrices
  - A. Aggarwal, M. M. Klawe, S. Moran, P. Shor, R. Wilber (1986)
  - $\Theta(n)$ speedup: $O(n^2)$ down to $O(n)$
- How are the two techniques related?

Quadrangle Inequality

Quadrangle Inequality

- Original Motivation
  Computing Optimal Binary Search Trees (Optimal BST)
  [Gilbert and Moore (1959)]

Quadrangle Inequality

- Original Motivation
  Computing Optimal Binary Search Trees (Optimal BST)
  [Gilbert and Moore (1959)]
- Optimal BST
  - Construct a search tree for n keys
  - n internal nodes corresponds to successful search
  - $n+1$ external nodes corresponds to unsuccessful search
  - Minimize the expected number of comparisons

Quadrangle Inequality

- Original Motivation

  Computing Optimal Binary Search Trees (Optimal BST)

  [Gilbert and Moore (1959)]

- Optimal BST
  - Construct a search tree for n keys
  - n internal nodes corresponds to successful search
  - $n+1$ external nodes corresponds to unsuccessful search
  - Minimize the expected number of comparisons

- Solution: Dynamic Programming

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

for some $w(i,j)$ that can be computed in $O(1)$ time.

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

    $n = 6$

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i < j) \\ 0 & (i = j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

  $n = 6$

	0	1	2	3	4	5	6
0	0						
1		0					
2			0				
3				0			
4					0		
5						0	
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\le j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

    $n = 6$

	0	1	2	3	4	5	6
0	0	230					
1		0	146				
2			0	75			
3				0	43		
4					0	44	
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i < j) \\ 0 & (i = j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

$n = 6$

	0	1	2	3	4	5	6
0	0	230	433				
1		0	146	260			
2			0	75	141		
3				0	43	119	
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

    $n = 6$

	0	1	2	3	4	5	6
0	0	230	433	586			
1		0	146	260	349		
2			0	75	141	250	
3				0	43	119	204
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

$n = 6$

	0	1	2	3	4	5	6
0	0	230	433	586	698		
1		0	146	260	349	491	
2			0	75	141	250	357
3				0	43	119	204
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\le j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

    $n = 6$

	0	1	2	3	4	5	6
0	0	230	433	586	698	862	
1		0	146	260	349	491	624
2			0	75	141	250	357
3				0	43	119	204
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\le j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example:

$n = 6$

	0	1	2	3	4	5	6
0	0	230	433	586	698	862	1002
1		0	146	260	349	491	624
2			0	75	141	250	357
3				0	43	119	204
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Standard Calculation

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - Diagonal by diagonal
  - An example: $n = 6$
  - Running time: $O(n^3)$

	0	1	2	3	4	5	6
0	0	230	433	586	698	862	1002
1		0	146	260	349	491	624
2			0	75	141	250	357
3				0	43	119	204
4					0	44	121
5						0	52
6							0

Quadrangle Inequality

- Speedup: $O(n^3) \to O(n^2)$ [Knuth (1971)]

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i < j) \\ 0 & (i = j) \end{cases}$$

Quadrangle Inequality

- Speedup: $O(n^3) \to O(n^2)$ [Knuth (1971)]

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i < j) \\ 0 & (i = j) \end{cases}$$

  - $K_B(i,j)$ the index t that achieves the minimum.

Quadrangle Inequality

- Speedup: $O(n^3) \to O(n^2)$ [Knuth (1971)]

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - $K_B(i,j)$ the index t that achieves the minimum.
  - Theorem in [Knuth (1971)]

$$K_B(i,j-1) \leq K_B(i,j) \leq K_B(i+1,j)$$

Quadrangle Inequality

- Speedup: $O(n^3) \to O(n^2)$ [Knuth (1971)]

$$B_{i,j} = \begin{cases} w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\} & (i<j) \\ 0 & (i=j) \end{cases}$$

  - $K_B(i,j)$ the index t that achieves the minimum.
  - Theorem in [Knuth (1971)]

$$K_B(i,j-1) \leq K_B(i,j) \leq K_B(i+1,j)$$

	$j-1$	j
i	$K_B(i,j-1)$	$K_B(i,j)$
$i+1$		$K_B(i+1,j)$

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \le K_B(i, j) \le K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0					
1			1				
2				2			
3					3		
4						4	
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1				
2				2			
3					3		
4						4	
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1	1			
2				2			
3					3		
4						4	
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1	1			
2				2	2		
3					3		
4						4	
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1	1			
2				2	2		
3					3	4	
4						4	
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1	1			
2				2	2		
3					3	4	
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \le K_B(i, j) \le K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0				
1			1	1			
2				2	2		
3					3	4	
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0	0			
1			1	1	1		
2				2	2	2	
3					3	4	4
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0	0	0		
1			1	1	1	1	
2				2	2	2	4
3					3	4	4
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0	0	0	1	
1			1	1	1	1	2
2				2	2	2	4
3					3	4	4
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0	0	0	1	1
1			1	1	1	1	2
2				2	2	2	4
3					3	4	4
4						4	5
5							5
6							

Quadrangle Inequality

- Speedup: $K_B(i, j-1) \leq K_B(i, j) \leq K_B(i+1, j)$
- The index table

	0	1	2	3	4	5	6
0		0	0	0	0	1	1
1			1	1	1	1	2
2				2	2	2	4
3					3	4	4
4						4	5
5							5
6							

- Running time: $O(n^3)$ down to $O(n^2)$

Quadrangle Inequality

Quadrangle Inequality

- Definition [Yao (1980, 1982)]

Quadrangle Inequality

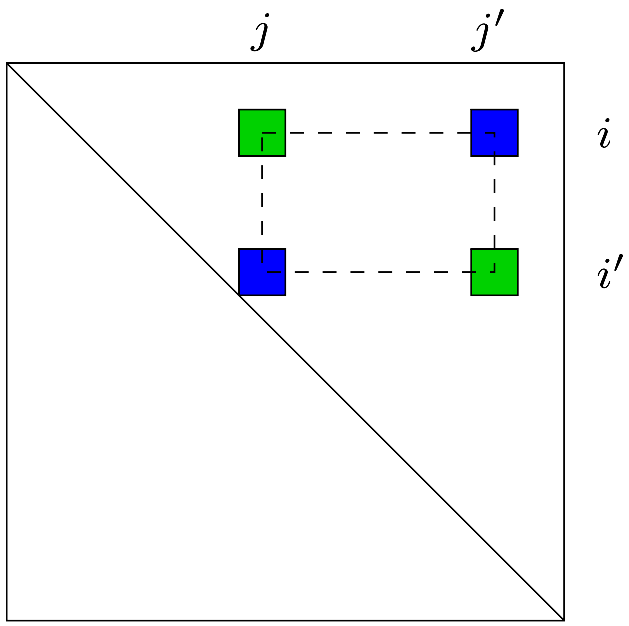
- Definition [Yao (1980, 1982)]
  - Function $f(i,j), (0 \le i \le j \le n)$
    satisfies a Quadrangle Inequality (QI), if $\forall i \le i' \le j \le j'$

$$f(i,j) + f(i',j') \le f(i',j) + f(i,j')$$

Quadrangle Inequality

- Definition [Yao (1980, 1982)]
  - Function $f(i,j), (0 \le i \le j \le n)$
    satisfies a Quadrangle Inequality (QI), if $\forall i \le i' \le j \le j'$

$$f(i,j) + f(i',j') \le f(i',j) + f(i,j')$$



Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

- Lemmas from [Yao (1980)]

Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

- Lemmas from [Yao (1980)]
  - (A) If $w(i,j)$ satisfies QI (and some additional constraints),
    $\Rightarrow B_{i,j}$ satisfies QI.

Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

- Lemmas from [Yao (1980)]
  - (A) If $w(i,j)$ satisfies QI (and some additional constraints),
    $\Rightarrow B_{i,j}$ satisfies QI.
  - (B) If $B_{i,j}$ satisfies QI,
    $\Rightarrow K_B(i,j-1) \leq K_B(i,j) \leq K_B(i+1,j)$

Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

- Lemmas from [Yao (1980)]
  - (A) If $w(i,j)$ satisfies QI (and some additional constraints),
    $\Rightarrow B_{i,j}$ satisfies QI.
  - (B) If $B_{i,j}$ satisfies QI,
    $\Rightarrow K_B(i,j-1) \leq K_B(i,j) \leq K_B(i+1,j)$
- In optimal BST problem,

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

Speedup using Quadrangle Inequality

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

- Lemmas from [Yao (1980)]
  - (A) If $w(i,j)$ satisfies QI (and some additional constraints),
    $\Rightarrow B_{i,j}$ satisfies QI.
  - (B) If $B_{i,j}$ satisfies QI,
    $\Rightarrow K_B(i,j-1) \leq K_B(i,j) \leq K_B(i+1,j)$

- In optimal BST problem,

$$B_{i,j} = w(i,j) + \min_{i<t\leq j}\{B_{i,t-1} + B_{t,j}\}$$

  - The specific $w(i,j)$ satisfies QI (and the additional constraints).

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Totally Monotone Matrices

- Definition

Totally Monotone Matrices

- Definition M is an $m \times n$ matrix

Totally Monotone Matrices

- Definition M is an $m \times n$ matrix
  - $\text{RM}_M(i)$ is index of minimum item of row i of M.

Totally Monotone Matrices

- Definition M is an $m \times n$ matrix
  - RM$_M(i)$ is index of minimum item of row i of M.
  - M is Monotone if $\forall i \leq i'$, RM$_M(i) \leq$ RM$_M(i')$.

Totally Monotone Matrices

- Definition M is an $m \times n$ matrix
  - $\text{RM}_M(i)$ is index of minimum item of row i of M.
  - M is Monotone if $\forall i \leq i'$, $\text{RM}_M(i) \leq \text{RM}_M(i')$.

7	2	4	3	8	9	$\text{RM}_M(1) = 2$
5	1	5	1	6	5	$\text{RM}_M(2) = 4$
7	1	2	0	3	1	$\text{RM}_M(3) = 4$
9	4	5	1	3	2	$\text{RM}_M(4) = 4$
8	4	5	3	4	3	$\text{RM}_M(5) = 6$
9	6	7	5	6	5	$\text{RM}_M(6) = 6$

Totally Monotone Matrices

- Definition M is an $m \times n$ matrix
  - RM$_M(i)$ is index of minimum item of row i of M.
  - M is Monotone if $\forall i \leq i'$, RM$_M(i) \leq$ RM$_M(i')$.

7	2	4	3	8	9	RM$_M(1) = 2$
5	1	5	1	6	5	RM$_M(2) = 4$
7	1	2	0	3	1	RM$_M(3) = 4$
9	4	5	1	3	2	RM$_M(4) = 4$
8	4	5	3	4	3	RM$_M(5) = 6$
9	6	7	5	6	5	RM$_M(6) = 6$

- An $m \times n$ matrix M is Totally Monotone (TM) if every 2×2 submatrix is Monotone.

SMAWK Algorithm

SMAWK Algorithm

- Motivation

  Find all m row minima of an implicitly given $m \times n$ matrix M

SMAWK Algorithm

- Motivation

  Find all m row minima of an implicitly given $m \times n$ matrix M

- Naive Algorithm: $O(mn)$

SMAWK Algorithm

- Motivation

  Find all m row minima of an implicitly given $m \times n$ matrix M

- Naive Algorithm: $O(mn)$

- SMAWK Algorithm

  [Aggarwal, Klawe, Moran, Shor, Wilber (1986)]

SMAWK Algorithm

- Motivation

  Find all m row minima of an implicitly given $m \times n$ matrix M

- Naive Algorithm: $O(mn)$

- SMAWK Algorithm

  [Aggarwal, Klawe, Moran, Shor, Wilber (1986)]

  - If M is Totally Monotone,

    all m row minima can be found in $O(m + n)$ time.

SMAWK Algorithm

- Motivation

  Find all m row minima of an implicitly given $m \times n$ matrix M

- Naive Algorithm: $O(mn)$

- SMAWK Algorithm

  [Aggarwal, Klawe, Moran, Shor, Wilber (1986)]

  - If M is Totally Monotone,

    all m row minima can be found in $O(m + n)$ time.

  - Usually $\Theta(n)$ speedup: $O(n^2)$ down to $O(n)$.

The Monge Property

The Monge Property

- Motivation

  TM property is often established via Monge property.

The Monge Property

- Motivation

  TM property is often established via Monge property.

- Definition

  An $m \times n$ matrix M is Monge if $\forall i \leq i'$ and $\forall j \leq j'$

$$M_{i,j} + M_{i',j'} \leq M_{i',j} + M_{i,j'}$$

The Monge Property

- Motivation

  TM property is often established via Monge property.

- Definition

  An $m \times n$ matrix M is Monge if $\forall i \leq i'$ and $\forall j \leq j'$

  $$M_{i,j} + M_{i',j'} \leq M_{i',j} + M_{i,j'}$$

- Theorems

  M is Monge $\Rightarrow$ M is Totally Monotone

  M is Monge $\not\Leftarrow$ M is Totally Monotone

The Monge Property

Quadrangle Inequality

Function $f(i,j)$

$\forall i \leq i' \leq j \leq j'$

$$f(i,j) + f(i',j') \leq f(i',j) + f(i,j')$$

Monge

Matrix M

$\forall i \leq i'$ and $\forall j \leq j'$

$$M_{i,j} + M_{i',j'} \leq M_{i',j} + M_{i,j'}$$

The Monge Property

Quadrangle Inequality

Function $f(i,j)$

$\forall i \le i' \le j \le j'$

$$f(i,j) + f(i',j') \le f(i',j) + f(i,j')$$

Monge

Matrix M

$\forall i \le i'$ and $\forall j \le j'$

$$M_{i,j} + M_{i',j'} \le M_{i',j} + M_{i,j'}$$

- QI vs. Monge

The Monge Property

Quadrangle Inequality

Function $f(i, j)$

$\forall i \leq i' \leq j \leq j'$

$$f(i, j) + f(i', j') \leq f(i', j) + f(i, j')$$

Monge

Matrix M

$\forall i \leq i'$ and $\forall j \leq j'$

$$M_{i,j} + M_{i',j'} \leq M_{i',j} + M_{i,j'}$$

- QI vs. Monge
  - Different names for same type of inequality.

The Monge Property

Quadrangle Inequality	Monge
Function $f(i,j)$	Matrix M
$\forall i \le i' \le j \le j'$	$\forall i \le i'$ and $\forall j \le j'$
$f(i,j) + f(i',j') \le f(i',j) + f(i,j')$	$M_{i,j} + M_{i',j'} \le M_{i',j} + M_{i,j'}$

- QI vs. Monge
  - Different names for same type of inequality.
  - Used differently in literature.

The Monge Property

Quadrangle Inequality

Function $f(i,j)$

$\forall i \leq i' \leq j \leq j'$

$f(i,j) + f(i',j') \leq f(i',j) + f(i,j')$

Monge

Matrix M

$\forall i \leq i'$ and $\forall j \leq j'$

$M_{i,j} + M_{i',j'} \leq M_{i',j} + M_{i,j'}$

- QI vs. Monge
  - Different names for same type of inequality.
  - Used differently in literature.
    - QI: $f(i,j)$ is function to be calculated.
    - Monge: $M_{i,j}$ implicitly given.

The Monge Property

Quadrangle Inequality

Function $f(i,j)$

$\forall i \le i' \le j \le j'$

$$f(i,j) + f(i',j') \le f(i',j) + f(i,j')$$

Monge

Matrix M

$\forall i \le i'$ and $\forall j \le j'$

$$M_{i,j} + M_{i',j'} \le M_{i',j} + M_{i,j'}$$

- QI vs. Monge
  - Different names for same type of inequality.
  - Used differently in literature.
    - QI: $f(i,j)$ is function to be calculated.
      Need all $f(i,j)$ entries.
    - Monge: $M_{i,j}$ implicitly given.
      Only need the row minima, but not other entries.

Relationship?

Quadrangle Inequality Totally Monotone (Monge)

Relationship?

Quadrangle Inequality	Totally Monotone (Monge)
A matrix to be calculated	A matrix given implicitly

Relationship?

Quadrangle Inequality

A matrix to be calculated

Need all $O(n^2)$ entries

Totally Monotone (Monge)

A matrix given implicitly

Need only $O(n)$ row minima

Relationship?

Quadrangle Inequality	Totally Monotone (Monge)
A matrix to be calculated	A matrix given implicitly
Need all $O(n^2)$ entries	Need only $O(n)$ row minima
$O(n^3)$ to $O(n^2)$ speedup	$O(n^2)$ to $O(n)$ speedup

Relationship?

Quadrangle Inequality

A matrix to be calculated

Need all $O(n^2)$ entries

$O(n^3)$ to $O(n^2)$ speedup

Totally Monotone (Monge)

A matrix given implicitly

Need only $O(n)$ row minima

$O(n^2)$ to $O(n)$ speedup

- This talk

Relationship?

Quadrangle Inequality	Totally Monotone (Monge)
A matrix to be calculated	A matrix given implicitly
Need all $O(n^2)$ entries	Need only $O(n)$ row minima
$O(n^3)$ to $O(n^2)$ speedup	$O(n^2)$ to $O(n)$ speedup

- This talk
  - QI instance is decomposed into $\Theta(n)$ TM instances

Relationship?

Quadrangle Inequality	Totally Monotone (Monge)
A matrix to be calculated	A matrix given implicitly
Need all $O(n^2)$ entries	Need only $O(n)$ row minima
$O(n^3)$ to $O(n^2)$ speedup	$O(n^2)$ to $O(n)$ speedup

- This talk
  - QI instance is decomposed into $\Theta(n)$ TM instances
  - Each TM instance requires $O(n)$ time

Relationship?

Quadrangle Inequality	Totally Monotone (Monge)
A matrix to be calculated	A matrix given implicitly
Need all $O(n^2)$ entries	Need only $O(n)$ row minima
$O(n^3)$ to $O(n^2)$ speedup	$O(n^2)$ to $O(n)$ speedup

- This talk
  - QI instance is decomposed into $\Theta(n)$ TM instances
  - Each TM instance requires $O(n)$ time
  - $\Rightarrow$ QI instance requires $O(n^2)$ time in total

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Decompositions

QI instance $\longrightarrow \Theta(n)$ TM instances

Decompositions

QI instance $\longrightarrow \Theta(n)$ TM instances

- D^d decomposition

- L^m and R^m decompositions

Decompositions

QI instance $\longrightarrow$ $\Theta(n)$ TM instances

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance
- L^m and R^m decompositions

Decompositions

QI instance $\longrightarrow$ $\Theta(n)$ TM instances

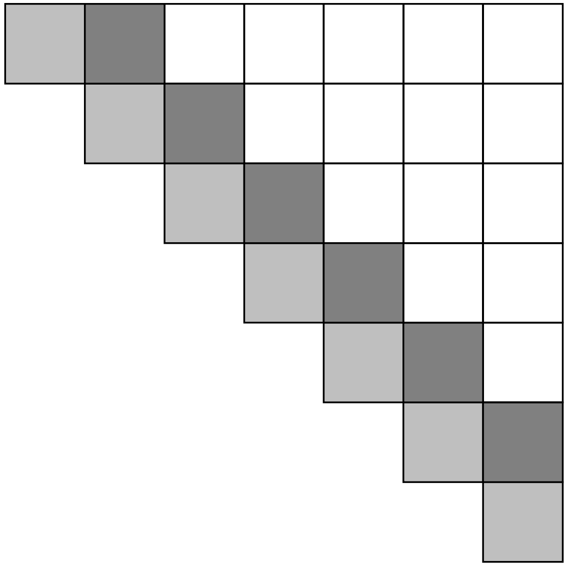
- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance
- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance

D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance

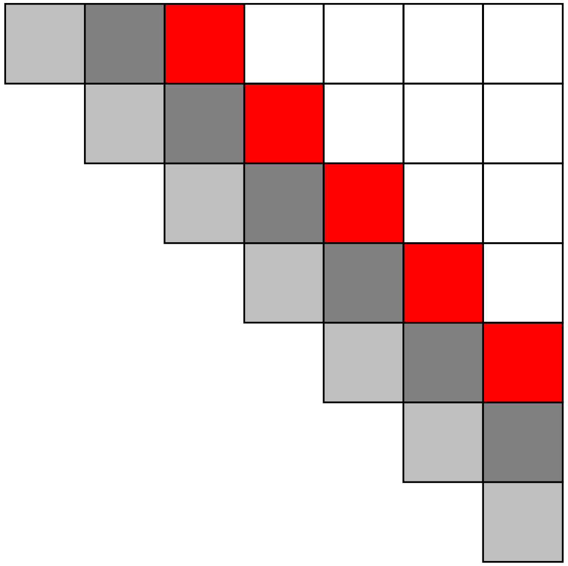
D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance



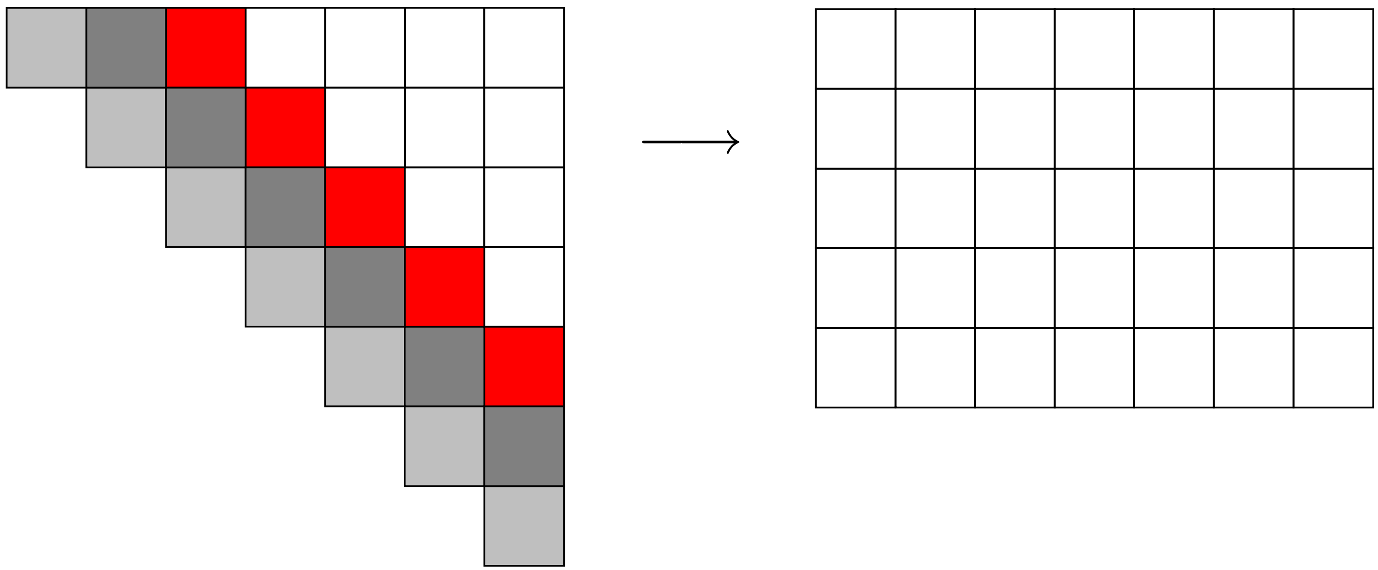
D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance



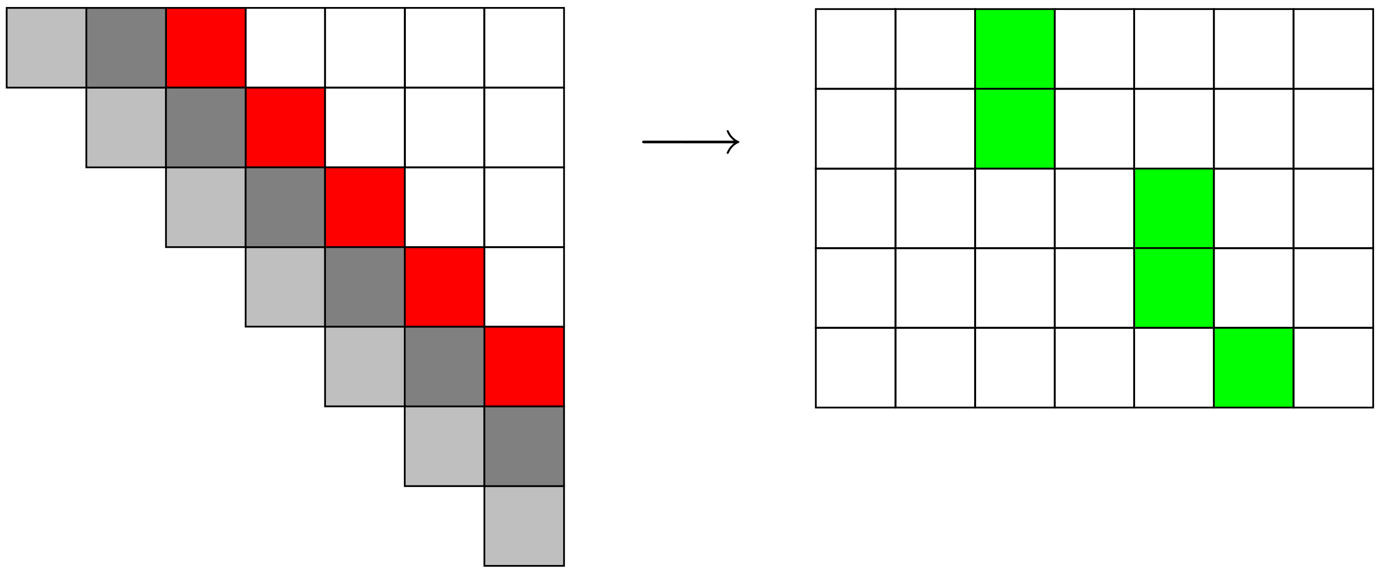
D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance



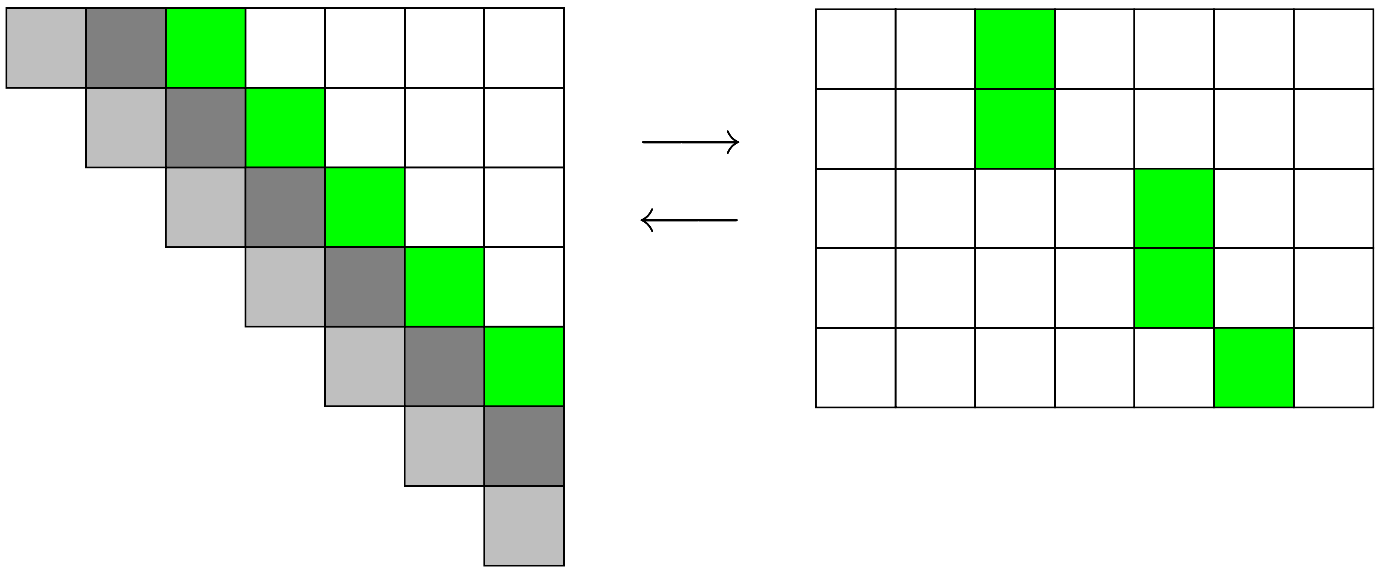
D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance



D^d Decomposition

- D^d decomposition
  - Each diagonal $\longrightarrow$ TM instance

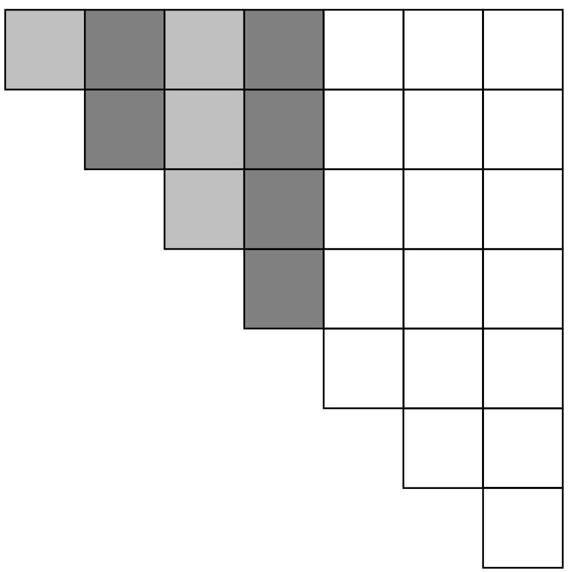


L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance

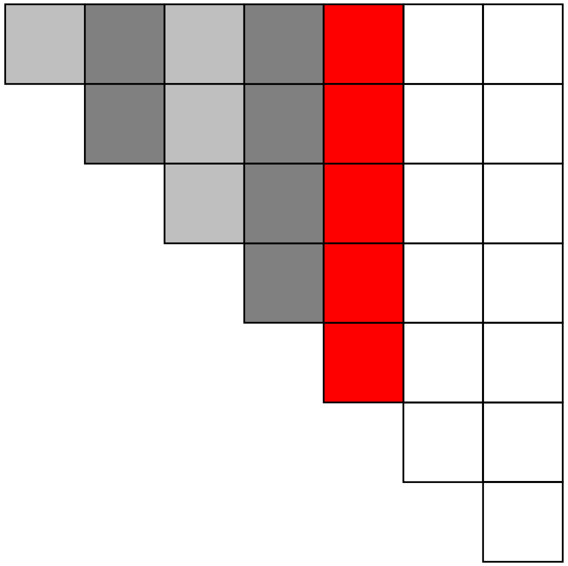
L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance



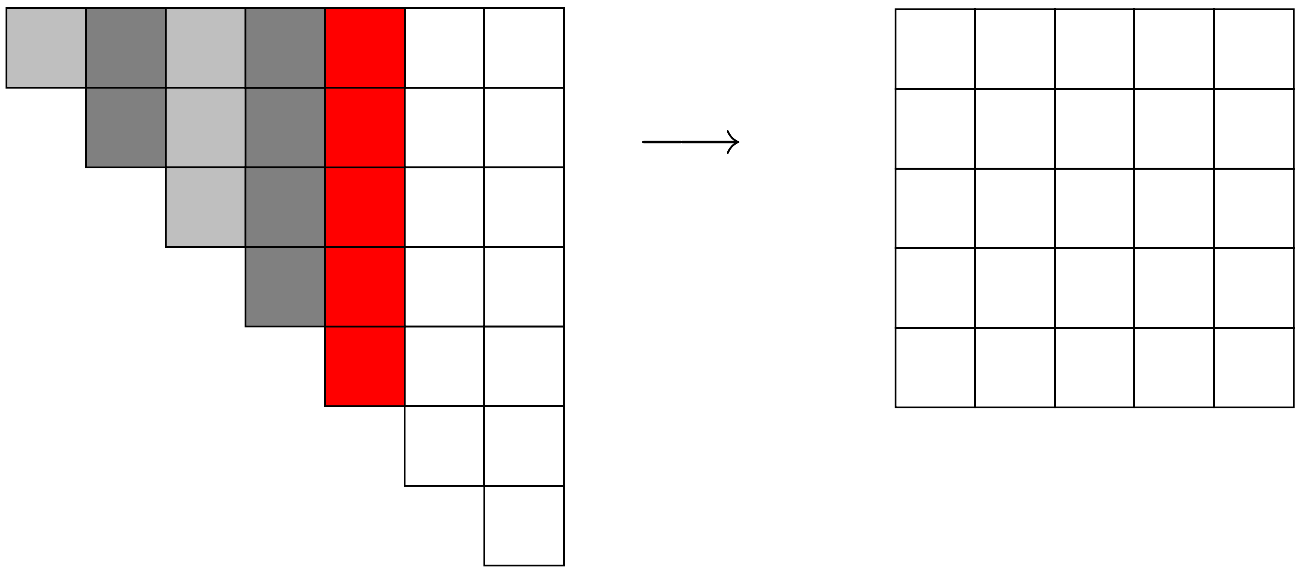
L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance



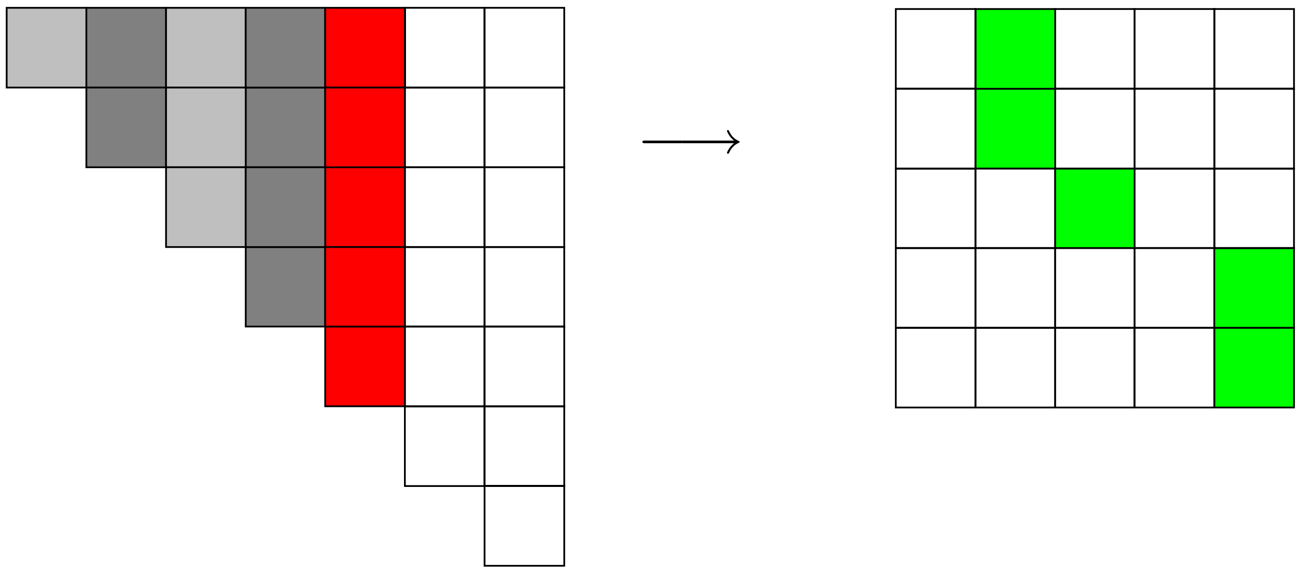
L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance



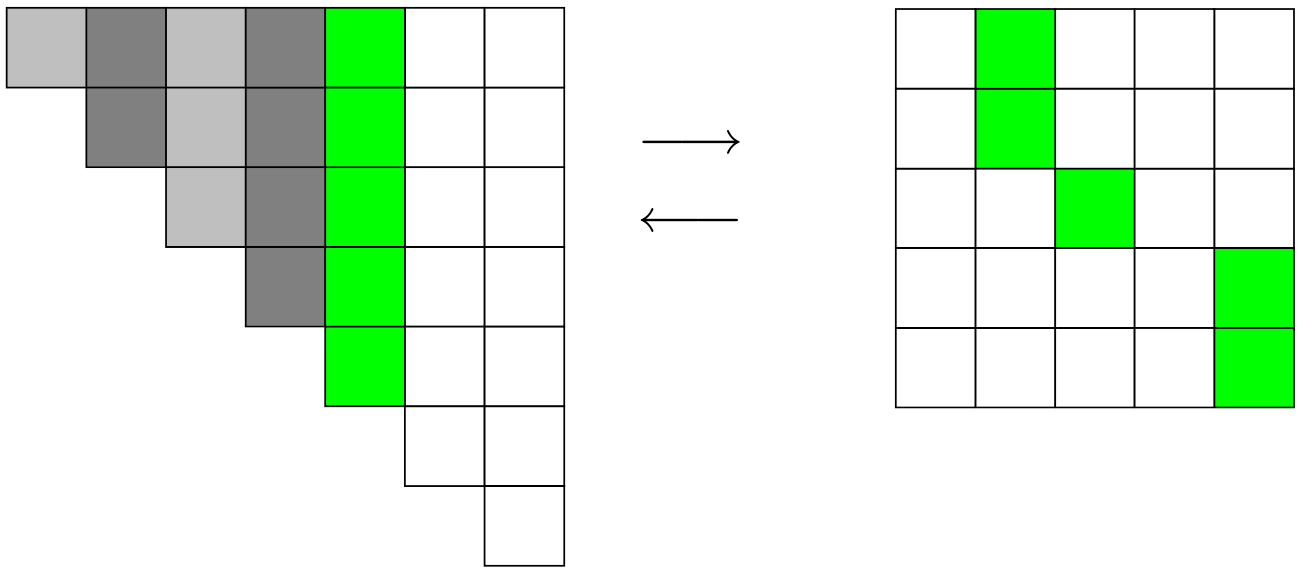
L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance



L^m and R^m Decompositions (R^m shown)

- L^m and R^m decompositions
  - L^m: Each row $\longrightarrow$ TM instance
  - R^m: Each column $\longrightarrow$ TM instance



D^d Decomposition

- Definition

D^d Decomposition

- Definition
  - For diagonal d, $(1 \leq d < n)$

    $B_{i,i+d} = w(i, i+d) + \min_{i<j\leq i+d}\{B_{i,j-1} + B_{j,i+d}\}$

D^d Decomposition

- Definition
  - For diagonal d, $(1 \leq d < n)$

    $$B_{i,i+d} = w(i,i+d) + \min_{i<j\leq i+d}\{B_{i,j-1} + B_{j,i+d}\}$$
  - Define $(n-d+1) \times (n+1)$ matrix D^d

    $$D^d_{i,j} = \begin{cases} w(i,i+d) + \{B_{i,j-1} + B_{j,i+d}\} & \text{if } 0 \leq i < j \leq i+d \leq n \\ \infty & \text{otherwise} \end{cases}$$

D^d Decomposition

- Definition
  - For diagonal d, $(1 \le d < n)$
    $$B_{i,i+d} = w(i, i+d) + \min_{i<j\le i+d}\{B_{i,j-1} + B_{j,i+d}\}$$
  - Define $(n-d+1) \times (n+1)$ matrix D^d
    $$D^d_{i,j} = \begin{cases} w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} & \text{if } 0 \le i < j \le i+d \le n \\ \infty & \text{otherwise} \end{cases}$$
  - Then, $B_{i,i+d} = \min_{0\le j\le n} D^d_{i,j}$ = minimum of row i of D^d

D^d Decomposition

- Definition
  - For diagonal d, $(1 \leq d < n)$

    $$B_{i,i+d} = w(i,i+d) + \min_{i<j\leq i+d}\{B_{i,j-1} + B_{j,i+d}\}$$
  - Define $(n-d+1) \times (n+1)$ matrix D^d

    $$D^d_{i,j} = \begin{cases} w(i,i+d) + \{B_{i,j-1} + B_{j,i+d}\} & \text{if } 0 \leq i < j \leq i+d \leq n \\ \infty & \text{otherwise} \end{cases}$$
  - Then, $B_{i,i+d} = \min_{0\leq j\leq n} D^d_{i,j}$ = minimum of row i of D^d
- Lemma

  D^d is Monge, for each $1 \leq d < n$.

D^d Decomposition

- Definition
  - For diagonal d, $(1 \le d < n)$

    $B_{i,i+d} = w(i, i+d) + \min_{i<j\le i+d}\{B_{i,j-1} + B_{j,i+d}\}$
  - Define $(n-d+1) \times (n+1)$ matrix D^d

$$D^d_{i,j} = \begin{cases} w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} & \text{if } 0 \le i < j \le i+d \le n \\ \infty & \text{otherwise} \end{cases}$$

  - Then, $B_{i,i+d} = \min_{0\le j\le n} D^d_{i,j}$ = minimum of row i of D^d
- Lemma

  D^d is Monge, for each $1 \le d < n$.
- For fixed d, SMAWK can be used to find all the $B_{i,i+d}$ in $O(n)$ time.
  - $\Rightarrow O(n^2)$ time for all D^d.

R^m Decomposition

- Definition

R^m Decomposition

- Definition
  - For column m, $(1 \leq m \leq n)$
    $B_{i,m} = w(i,m) + \min_{i<j\leq m}\{B_{i,j-1} + B_{j,m}\}$

R^m Decomposition

- Definition
  - For column m, $(1 \leq m \leq n)$

    $$B_{i,m} = w(i,m) + \min_{i<j\leq m}\{B_{i,j-1} + B_{j,m}\}$$

  - Define $(m+1) \times (m+1)$ matrix R^m

    $$R^m_{i,j} = \begin{cases} w(i,m) + \{B_{i,j-1} + B_{j,m}\} & \text{if } 0 \leq i < j \leq m \\ \infty & \text{otherwise} \end{cases}$$

R^m Decomposition

- Definition
  - For column m, $(1 \leq m \leq n)$

    $$B_{i,m} = w(i,m) + \min_{i<j\leq m}\{B_{i,j-1} + B_{j,m}\}$$
  - Define $(m+1) \times (m+1)$ matrix R^m

    $$R^m_{i,j} = \begin{cases} w(i,m) + \{B_{i,j-1} + B_{j,m}\} & \text{if } 0 \leq i < j \leq m \\ \infty & \text{otherwise} \end{cases}$$
  - Then, $B_{i,m} = \min_{0<j\leq m} R^m_{i,j}$

R^m Decomposition

- Definition
  - For column m, $(1 \leq m \leq n)$

    $$B_{i,m} = w(i,m) + \min_{i<j\leq m}\{B_{i,j-1} + B_{j,m}\}$$

  - Define $(m+1) \times (m+1)$ matrix R^m

    $$R^m_{i,j} = \begin{cases} w(i,m) + \{B_{i,j-1} + B_{j,m}\} & \text{if } 0 \leq i < j \leq m \\ \infty & \text{otherwise} \end{cases}$$

  - Then, $B_{i,m} = \min_{0<j\leq m} R^m_{i,j}$
- Lemma

  R^m is Monge, for each $1 \leq m \leq n$.

LARSCH Algorithm

LARSCH Algorithm

- D^d decomposition

LARSCH Algorithm

- D^d decomposition
  - $D^d_{i,j} = w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} \qquad (0 \le i < j \le i+d \le n)$
  - SMAWK algorithm

LARSCH Algorithm

- D^d decomposition
  - $D^d_{i,j} = w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} \qquad (0 \le i < j \le i+d \le n)$
  - SMAWK algorithm
- L^m and R^m decomposition
  - $R^m_{i,j} = w(i, m) + \{B_{i,j-1} + B_{j,m}\} \qquad (0 \le i < j \le m)$

LARSCH Algorithm

- D^d decomposition
  - $D^d_{i,j} = w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} \qquad (0 \le i < j \le i+d \le n)$
  - SMAWK algorithm
- L^m and R^m decomposition
  - $R^m_{i,j} = w(i, m) + \{B_{i,j-1} + B_{j,m}\} \qquad (0 \le i < j \le m)$
  - Can not use SMAWK algorithm:

    $B_{j,m}$ is row minimum of row j of R^m and is therefore not known.

LARSCH Algorithm

- D^d decomposition
  - $D^d_{i,j} = w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} \qquad (0 \le i < j \le i+d \le n)$
  - SMAWK algorithm
- L^m and R^m decomposition
  - $R^m_{i,j} = w(i, m) + \{B_{i,j-1} + B_{j,m}\} \qquad (0 \le i < j \le m)$
  - Can not use SMAWK algorithm:
    $B_{j,m}$ is row minimum of row j of R^m and is therefore not known.
  - LARSCH algorithm [Larmore, Schieber (1990)]
    permits calculating row minima of TM matrices in $O(n)$ time,
    even with this dependency.

LARSCH Algorithm

- D^d decomposition
  - $D^d_{i,j} = w(i, i+d) + \{B_{i,j-1} + B_{j,i+d}\} \qquad (0 \le i < j \le i+d \le n)$
  - SMAWK algorithm
- L^m and R^m decomposition
  - $R^m_{i,j} = w(i, m) + \{B_{i,j-1} + B_{j,m}\} \qquad (0 \le i < j \le m)$
  - Can not use SMAWK algorithm:
    $B_{j,m}$ is row minimum of row j of R^m and is therefore not known.
  - LARSCH algorithm [Larmore, Schieber (1990)]
    permits calculating row minima of TM matrices in $O(n)$ time,
    even with this dependency.
  - $O(n)$ time for each column $\Rightarrow O(n^2)$ in total.

LARSCH Algorithm

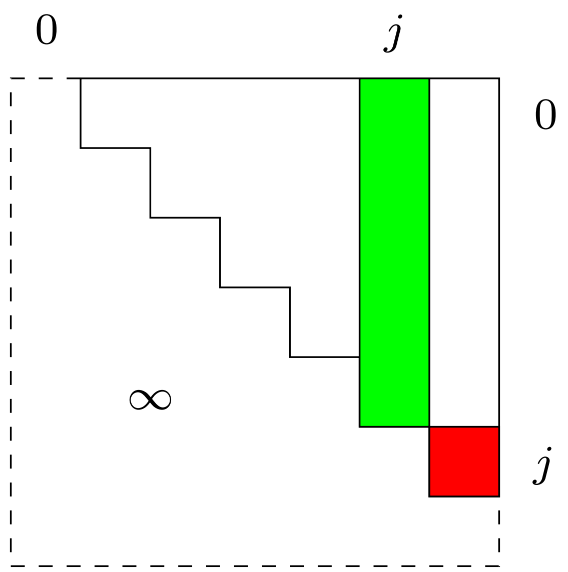
Finding row minima in totally monotone matrices with limited dependency.

LARSCH Algorithm

Finding row minima in totally monotone matrices with limited dependency.

Entries of column j can depend on the row minima of rows i where $M_{i,j} = \infty$.

Green: the column j.
Red: rows that column j can depend on.



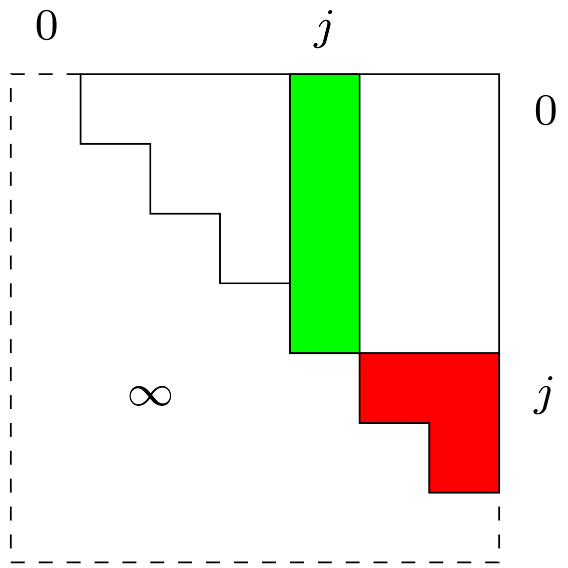
LARSCH Algorithm

Finding row minima in totally monotone matrices with limited dependency.

Entries of column j can depend on the row minima of rows i where $M_{i,j} = \infty$.

Green: the column j.

Red: rows that column j can depend on.



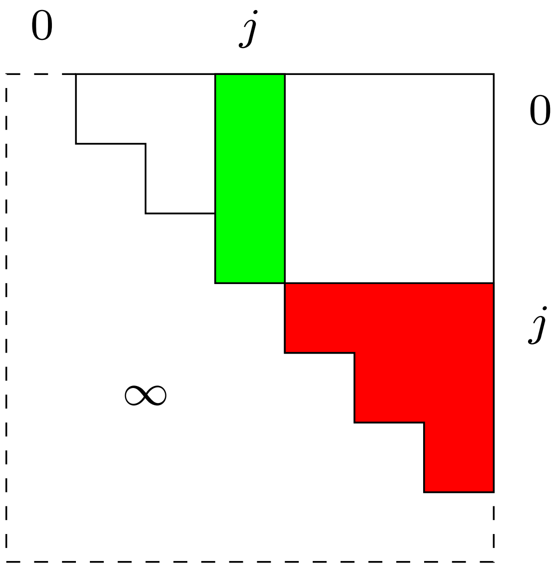
LARSCH Algorithm

Finding row minima in totally monotone matrices with limited dependency.

Entries of column j can depend on the row minima of rows i where $M_{i,j} = \infty$.

Green: the column j.

Red: rows that column j can depend on.



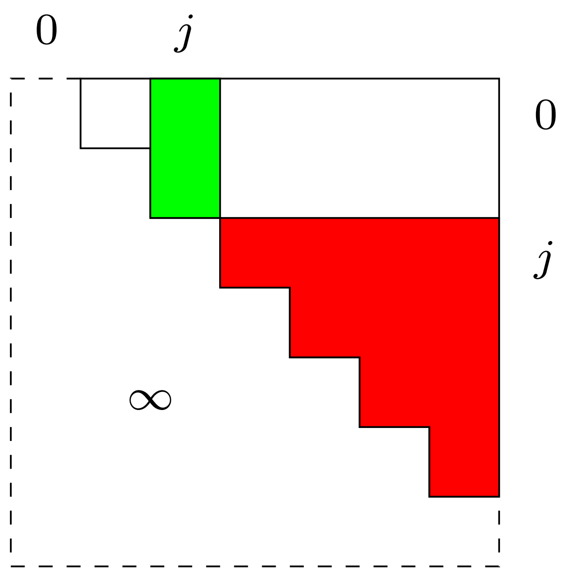
LARSCH Algorithm

Finding row minima in totally monotone matrices with limited dependency.

Entries of column j can depend on the row minima of rows i where $M_{i,j} = \infty$.

Green: the column j.

Red: rows that column j can depend on.

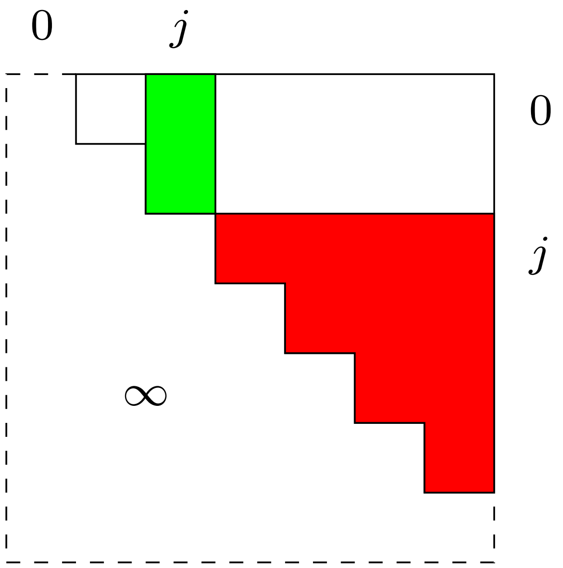


LARSCH Algorithm

Finding row minima in totally monotone matrices with limited dependency.

Entries of column j can depend on the row minima of rows i where $M_{i,j} = \infty$.

Green: the column j.
Red: rows that column j can depend on.



R^m satisfies the condition of LARSCH.

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Online Problem

Online Problem

- Definition: Two-sided online problem

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for $\text{Key}_l, \ldots, \text{Key}_r$

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for $\text{Key}_l, \ldots, \text{Key}_r$
  - Next step: Add either Key_{l-1} or Key_{r+1}.

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for $\text{Key}_l, \ldots, \text{Key}_r$
  - Next step: Add either Key_{l-1} or Key_{r+1}.
- An example

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for Key$_l, \ldots,$ Key$_r$
  - Next step: Add either Key$_{l-1}$ or Key$_{r+1}$.
- An example

  Input $=$ (Key$_l, \ldots,$ Key$_r$)

	1	2	3	4	5	6
1						
2		0	75	141	250	
3			0	43	119	
4				0	44	
5					0	
6						

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for $\text{Key}_l, \ldots, \text{Key}_r$
  - Next step: Add either Key_{l-1} or Key_{r+1}.
- An example

  Input $= ($ $\text{Key}_l, \ldots, \text{Key}_r, \text{Key}_{r+1})$

	1	2	3	4	5	6
1						
2		0	75	141	250	357
3			0	43	119	204
4				0	44	121
5					0	52
6						0

Online Problem

- Definition: Two-sided online problem
  - Current step: Optimal BST for $\text{Key}_l, \ldots, \text{Key}_r$
  - Next step: Add either Key_{l-1} or Key_{r+1}.
- An example

  Input $= (\text{Key}_{l-1}, \text{Key}_l, \ldots, \text{Key}_r, \text{Key}_{r+1})$

	1	2	3	4	5	6
1	0	146	260	349	491	624
2		0	75	141	250	357
3			0	43	119	204
4				0	44	121
5					0	52
6						0

Online Algorithm

	1	2	3	4	5	6
1	0	146	260	349	491	624
2		0	75	141	250	357
3			0	43	119	204
4				0	44	121
5					0	52
6						0

- Using L^m and R^m decomposition

Online Algorithm

	1	2	3	4	5	6
1	0	146	260	349	491	624
2		0	75	141	250	357
3			0	43	119	204
4				0	44	121
5					0	52
6						0

- Using L^m and R^m decomposition
  - $O(n)$ time worst case per step.

Outline

- Background
  - Kunth-Yao (KY) Quadrangle Inequality (QI) Speedup
  - SMAWK Algorithm for finding
    Row Minima of Totally Monotone (TM) Matrices
- The D^d Decomposition
  A transformation from QI to TM such that
  SMAWK solves KY problem as quickly as KY.
- The L^m and R^m Decompositions
  Another transformation from QI to TM that
  (1) implies KY speedup and (2) enables online solution.

Questions?