Tightening I/O Lower Bounds through the Hourglass Dependency Pattern

Lionel Eyraud-Dubois (Inria Bordeaux)
Guillaume Iooss (Inria Grenoble)
Julien Langou (University of Colorado Denver + Inria Lyon)
Fabrice Rastello (Inria Grenoble)

SPAA'24

18 June 2024

Motivation

- When optimizing for performance, many aspects to consider.
- Need to estimate some key program properties:
  - Volume of computation ?

    $\Rightarrow$ Algorithmic complexity.

Motivation

- When optimizing for performance, many aspects to consider.
- Need to estimate some key program properties:
  - Volume of computation ?

    $\Rightarrow$ Algorithmic complexity.
  - Volume of I/O to be transferred across memories ?

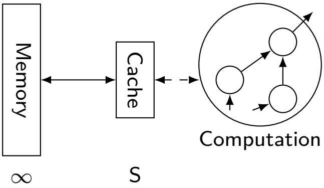
    $\Rightarrow$ **I/O Complexity**: minimal amount of I/O required.

Motivation

- When optimizing for performance, many aspects to consider.
- Need to estimate some key program properties:
  - Volume of computation ?
    $\Rightarrow$ Algorithmic complexity.
  - Volume of I/O to be transferred across memories ?
    $\Rightarrow$ **I/O Complexity**: minimal amount of I/O required.
- How to model & compute this I/O Complexity ?

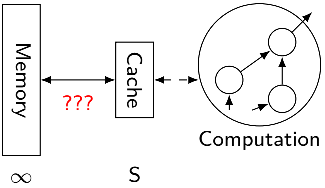
I/O Complexity

- 2-level memory model:



I/O Complexity

- 2-level memory model:

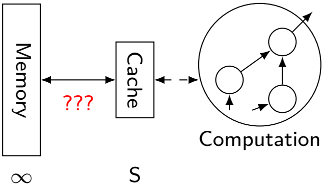


I/O Complexity of a program

Minimal number of memory transfer, **for any** schedule

I/O Complexity

- 2-level memory model:



I/O Complexity of a program

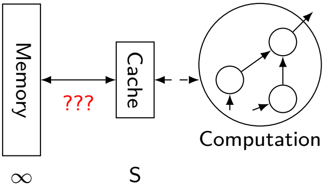
Minimal number of memory transfer, **for any** schedule

- Direct computation not feasible

  $\Rightarrow$ **Lower bound (proof)** + upper bound (exhibit schedule)

I/O Complexity

- 2-level memory model:



I/O Complexity of a program

Minimal number of memory transfer, **for any** schedule

- Direct computation not feasible
  - $\Rightarrow$ **Lower bound (proof)** + upper bound (exhibit schedule)
- Focus on Reads + No recomputation

Content of this presentation

- Background: **K-partitioning** proof method.

Content of this presentation

- Background: **K-partitioning** proof method.

- Why this is not optimal for some kernels?
  Identify pattern of dependence that causes this issue.
  $\Rightarrow$ **Hourglass pattern**.

Content of this presentation

- Background: **K-partitioning** proof method.
- Why this is not optimal for some kernels?
  Identify pattern of dependence that causes this issue.
  $\Rightarrow$ **Hourglass pattern**.
- Adapt K-partitioning to improve the bound.
  Integrated in automatic lower bound derivation tool (IOLB).
  $\Rightarrow$ Improve the bounds of many kernels by asymptotic factor.

Computational Directed Acyclic Graph

We need to reason about the computation of a program.

- **Computational Directed Acyclic Graph (CDAG)**:
  - Node = one computation, or input.
  - Edge = dependence between computations.

Computational Directed Acyclic Graph

We need to reason about the computation of a program.

- **Computational Directed Acyclic Graph (CDAG)**:
  - Node = one computation, or input.
  - Edge = dependence between computations.

- Needs regularity in a CDAG: **polyhedral programs**.
  - Loop indexes satisfies affine constraints (ex: "$0 \leq i < N$").
  - Memory accesses are affine (ex: "$A[2i - j + 1]$").

  $\Rightarrow$ Many linear algebra kernels fits these criteria.

K-partitioning method

Definition (K-set)

Set of nodes of the CDAG, such that the size of its *inset* (input data) is $\leq K$.

- **Idea:** Partition the CDAG into convex K-sets ($= K$-partition)

K-partitioning method

Definition (K-set)

Set of nodes of the CDAG, such that the size of its *inset* (input data) is $\leq K$.

- **Idea:** Partition the CDAG into convex K-sets ($= K$-partition)

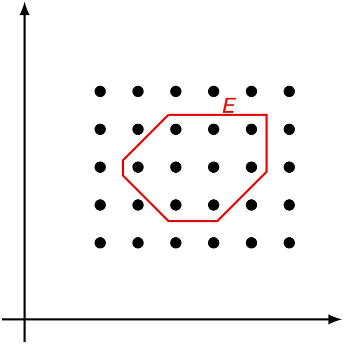
Theorem (Hong and Kung'81)

With S the cache size, for all K-partition:

$$\begin{aligned} \#I/O \geq\ & (K - S) \times min(Num\_KSets\_in\_KPartition) \\ \geq\ & (K - S) \times \frac{Num\_Nodes\_CDAG}{max(Size\_KSet)} \end{aligned}$$

$\Rightarrow$ Convert upper bound on K-set into lower bound on I/O.

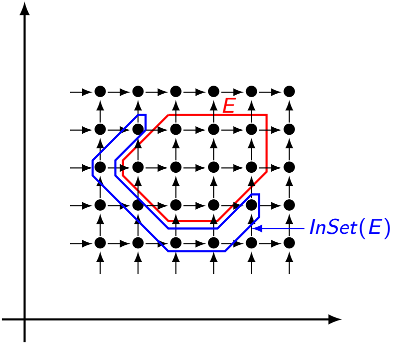
Deriving an upper bound of a K-set



E K-set of arbitrary shape

Upper bound on $|E|$?

Deriving an upper bound of a K-set



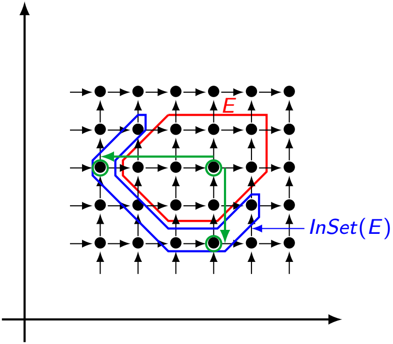
$\color{red}{E}$ K-set of arbitrary shape

Upper bound on $|E|$?

$\color{blue}{InSet(E)}$: input data of E

$|InSet(E)| \leq K$

Deriving an upper bound of a K-set



E K-set of arbitrary shape

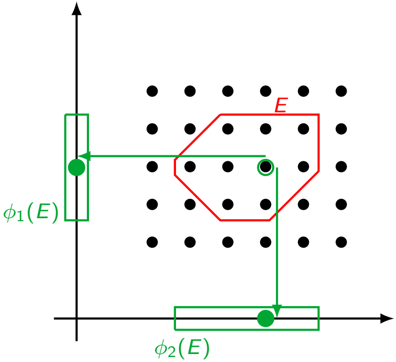
Upper bound on $|E|$?

$InSet(E)$: input data of E

$|InSet(E)| \leq K$

1) Derive paths that maps from E to $InSet(E)$

Deriving an upper bound of a K-set



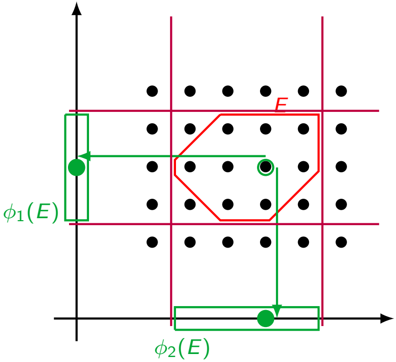
E K-set of arbitrary shape
Upper bound on $|E|$?

$InSet(E)$: input data of E
$|InSet(E)| \leq K$

1) Derive paths that maps from E to $InSet(E)$

2) Projections ϕ_x from paths
$|\phi_x(E)| \leq |InSet(E)| \leq K$

Deriving an upper bound of a K-set



E K-set of arbitrary shape
Upper bound on $|E|$?

$InSet(E)$: input data of E
$|InSet(E)| \leq K$

1) Derive paths that maps from E to $InSet(E)$

2) Projections ϕ_x from paths
$|\phi_x(E)| \leq |InSet(E)| \leq K$

3) Brascamp-Lieb theorem:
$|E| \leq |\phi_1(E)| \times |\phi_2(E)|$
$\Rightarrow |E| \leq K^2$

Example: Modified Gram-Schmidt

```
for (k=0; k<N; k++) {
    nrm = 0.0;
    for (i=0; i<M; i++)
      nrm += A[i][k] * A[i][k];
    R[k][k] = sqrt(nrm);
    for (i = 0; i < M; i++)
      Q[i][k] = A[i][k] / R[k][k];
    for (j = k + 1; j < N; j++) {
      R[k][j] = 0.0;
      for (i = 0; i < M; i++)
SR:     R[k][j] += Q[i][k] * A[i][j];
      for (i = 0; i < M; i++)
SA:     A[i][j] = A[i][j] - Q[i][k] * R[k][j];
    }
}
```

Example: Modified Gram-Schmidt

```
  for (k=0; k<N; k++) {
     nrm = 0.0;
     for (i=0; i<M; i++)
        nrm += A[i][k] * A[i][k];
     R[k][k] = sqrt(nrm);
     for (i = 0; i < M; i++)
        Q[i][k] = A[i][k] / R[k][k];
     for (j = k + 1; j < N; j++) {
        R[k][j] = 0.0;
        for (i = 0; i < M; i++)
SR:        R[k][j] += Q[i][k] * A[i][j];
        for (i = 0; i < M; i++)
SA:        A[i][j] = A[i][j] - Q[i][k] * R[k][j];
     }
  }
```

$$A[i][j] = \underbrace{A[i][j]}_{\phi_{i,j}} - \underbrace{Q[i][k]}_{\phi_{i,k}} * \underbrace{R[k][j]}_{\phi_{k,j}};$$

3 paths $\Rightarrow$ 3 projections:

$$|\phi_{\bullet,\bullet}(E)| \leq K$$

Example: Modified Gram-Schmidt

```
   for (k=0; k<N; k++) {
       nrm = 0.0;
       for (i=0; i<M; i++)
          nrm += A[i][k] * A[i][k];
       R[k][k] = sqrt(nrm);
       for (i = 0; i < M; i++)
          Q[i][k] = A[i][k] / R[k][k];
       for (j = k + 1; j < N; j++) {
          R[k][j] = 0.0;
          for (i = 0; i < M; i++)
SR:          R[k][j] += Q[i][k] * A[i][j];
          for (i = 0; i < M; i++)
SA:          A[i][j] = A[i][j] - Q[i][k] * R[k][j];
       }
   }
```

$$A[i][j] = \underbrace{A[i][j]}_{\phi_{i,j}} - \underbrace{Q[i][k]}_{\phi_{i,k}} * \underbrace{R[k][j]}_{\phi_{k,j}};$$

3 paths $\Rightarrow$ 3 projections:

$$|\phi_{\bullet,\bullet}(E)| \leq K$$

Brascamp-Lieb:

$$|E| \leq |\phi_{i,j}(E)|^{\frac{1}{2}} \times |\phi_{i,k}(E)|^{\frac{1}{2}} \times |\phi_{k,j}(E)|^{\frac{1}{2}}$$

$$\Rightarrow |E| \leq K^{\frac{3}{2}}$$

Example: Modified Gram-Schmidt

```
for (k=0; k<N; k++) {
    nrm = 0.0;
    for (i=0; i<M; i++)
       nrm += A[i][k] * A[i][k];
    R[k][k] = sqrt(nrm);
    for (i = 0; i < M; i++)
       Q[i][k] = A[i][k] / R[k][k];
    for (j = k + 1; j < N; j++) {
       R[k][j] = 0.0;
       for (i = 0; i < M; i++)
SR:       R[k][j] += Q[i][k] * A[i][j];
       for (i = 0; i < M; i++)
SA:       A[i][j] = A[i][j] - Q[i][k] * R[k][j];
    }
}
```

$$A[i][j] = \underbrace{A[i][j]}_{\phi_{i,j}} - \underbrace{Q[i][k]}_{\phi_{i,k}} * \underbrace{R[k][j]}_{\phi_{k,j}};$$

3 paths ⇒ 3 projections:

$$|\phi_{\bullet,\bullet}(E)| \leq K$$

Brascamp-Lieb:

$$|E| \leq |\phi_{i,j}(E)|^{\frac{1}{2}} \times |\phi_{i,k}(E)|^{\frac{1}{2}} \times |\phi_{k,j}(E)|^{\frac{1}{2}}$$

$$\Rightarrow |E| \leq K^{\frac{3}{2}}$$

$$Q_{MGS} \geq \Omega\left(\frac{MN^2}{\sqrt{S}}\right)$$

Example: Modified Gram-Schmidt

```
for (k=0; k<N; k++) {
    nrm = 0.0;
    for (i=0; i<M; i++)
       nrm += A[i][k] * A[i][k];
    R[k][k] = sqrt(nrm);
    for (i = 0; i < M; i++)
       Q[i][k] = A[i][k] / R[k][k];
    for (j = k + 1; j < N; j++) {
       R[k][j] = 0.0;
       for (i = 0; i < M; i++)
SR:       R[k][j] += Q[i][k] * A[i][j];
       for (i = 0; i < M; i++)
SA:       A[i][j] = A[i][j] - Q[i][k] * R[k][j];
    }
}
```

$$A[i][j] = \underbrace{A[i][j]}_{\phi_{i,j}} - \underbrace{Q[i][k]}_{\phi_{i,k}} * \underbrace{R[k][j]}_{\phi_{k,j}};$$

3 paths $\Rightarrow$ 3 projections:

$$|\phi_{\bullet,\bullet}(E)| \leq K$$

Brascamp-Lieb:

$$|E| \leq |\phi_{i,j}(E)|^{\frac{1}{2}} \times |\phi_{i,k}(E)|^{\frac{1}{2}} \times |\phi_{k,j}(E)|^{\frac{1}{2}}$$

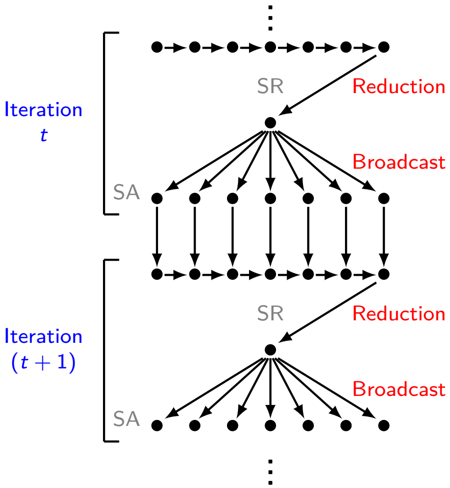
$$\Rightarrow |E| \leq K^{\frac{3}{2}}$$

$$Q_{MGS} \geq \Omega\left(\frac{MN^2}{\sqrt{S}}\right)$$

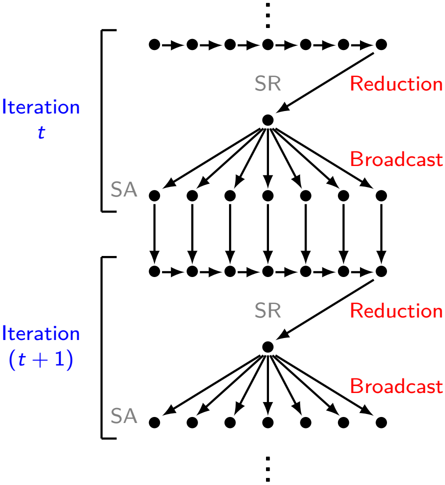
- Similar to the bound of a matrix multiplication...
  ...but best known I/O cost for MGS is: $O(MN^2)$. [Demmel12]

  $\Rightarrow$ Can we do better?

The Hourglass pattern



The Hourglass pattern



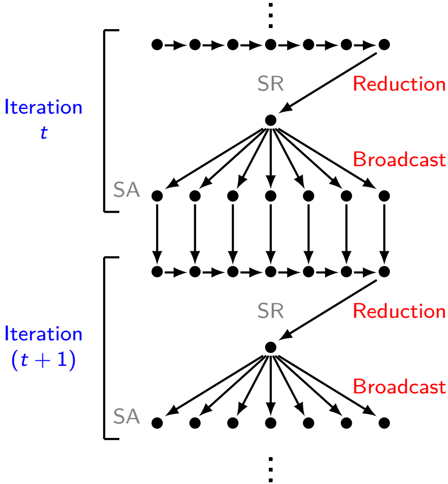
Constraints:

- Time dim (often outer)
- Broadcast/Red dim (often inner)
- Other dims: neutral

$+$

Width hourglass parametric (ex: M)

The Hourglass pattern



Constraints:

- Time dim (often outer)
- Broadcast/Red dim (often inner)
- Other dims: neutral

$+$

Width hourglass parametric (ex: M)

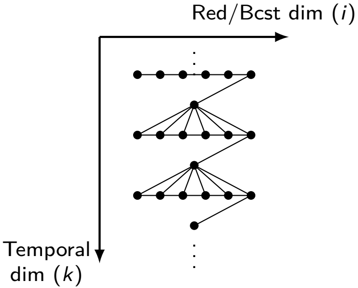
If reduction large, not tilable !

$\Rightarrow$ Strongly constraints shape of a (convex) K-set.

Implication on the shape of E

Split the connected components of E (a K-set) in 2: $E = E_1 \uplus E_2$

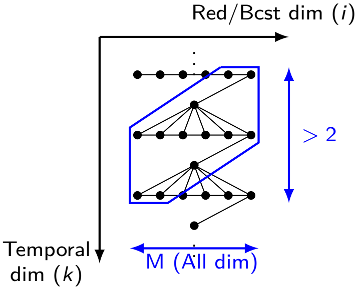
- Thick along temporal dimension (E_1)
  - $\Rightarrow$ Must cover all the Red/Bcst dim
- Flat along temporal dimension (E_2)



Implication on the shape of E

Split the connected components of E (a K-set) in 2: $E = E_1 \uplus E_2$

- Thick along temporal dimension (E_1)
  - $\Rightarrow$ Must cover all the Red/Bcst dim
- Flat along temporal dimension (E_2)



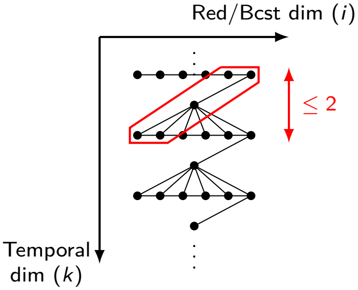
$|\phi_i(E_1)| = M$

If $\phi_{\bullet,i}$ is one projection:
$|\phi_\bullet(E_1)| \leq \frac{K}{M}$

Implication on the shape of E

Split the connected components of E (a K-set) in 2: $E = E_1 \uplus E_2$

- Thick along temporal dimension (E_1)
  - $\Rightarrow$ Must cover all the Red/Bcst dim
- Flat along temporal dimension (E_2)



$|\phi_i(E_1)| = M$

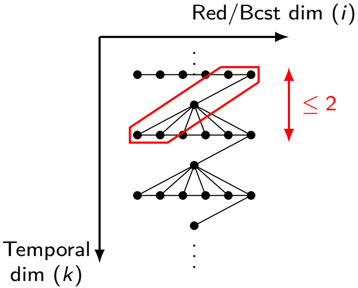
If $\phi_{\bullet,i}$ is one projection:
$|\phi_\bullet(E_1)| \leq \frac{K}{M}$

$|\phi_k(E_2)| \leq 2$

Implication on the shape of E

Split the connected components of E (a K-set) in 2: $E = E_1 \uplus E_2$

- Thick along temporal dimension (E_1)
  - $\Rightarrow$ Must cover all the Red/Bcst dim
- Flat along temporal dimension (E_2)



$|\phi_i(E_1)| = M$

If $\phi_{\bullet,i}$ is one projection: $|\phi_\bullet(E_1)| \leq \frac{K}{M}$

$|\phi_k(E_2)| \leq 2$

$\Rightarrow$ New bounds on projection sizes to exploit, on both parts.

Putting things together

Example - Modified Gram-Schmidt.

By adapting the list of projections given to Brascamp-Lieb:

- First part (Thick):
  - Instead of: $|E_1| \leq |\phi_{i,j}(E_1)|^{\frac{1}{2}} \times |\phi_{i,k}(E_1)|^{\frac{1}{2}} \times |\phi_{j,k}(E_1)|^{\frac{1}{2}} \leq K^{\frac{3}{2}}$.
  - We have: $|E_1| \leq |\phi_i(E_1)| \times |\phi_j(E_1)| \times |\phi_k(E_1)| \leq M \times \frac{K}{M} \times \frac{K}{M} = \frac{K^2}{M}$.

Putting things together

Example - Modified Gram-Schmidt.

By adapting the list of projections given to Brascamp-Lieb:

- First part (Thick):
  - Instead of: $|E_1| \leq |\phi_{i,j}(E_1)|^{\frac{1}{2}} \times |\phi_{i,k}(E_1)|^{\frac{1}{2}} \times |\phi_{j,k}(E_1)|^{\frac{1}{2}} \leq K^{\frac{3}{2}}$.
  - We have: $|E_1| \leq |\phi_i(E_1)| \times |\phi_j(E_1)| \times |\phi_k(E_1)| \leq M \times \frac{K}{M} \times \frac{K}{M} = \frac{K^2}{M}$.
- Second part (Flat):
  - Instead of: $|E_2| \leq K^{\frac{3}{2}}$.
  - We have: $|E_2| \leq |\phi_k(E_2)| \times |\phi_{i,j}(E_2)| \leq 2K$.

Putting things together

Example - Modified Gram-Schmidt.

By adapting the list of projections given to Brascamp-Lieb:

- First part (Thick):
  - Instead of: $|E_1| \leq |\phi_{i,j}(E_1)|^{\frac{1}{2}} \times |\phi_{i,k}(E_1)|^{\frac{1}{2}} \times |\phi_{j,k}(E_1)|^{\frac{1}{2}} \leq K^{\frac{3}{2}}$.
  - We have:
    $|E_1| \leq |\phi_i(E_1)| \times |\phi_j(E_1)| \times |\phi_k(E_1)| \leq M \times \frac{K}{M} \times \frac{K}{M} = \frac{K^2}{M}$.
- Second part (Flat):
  - Instead of: $|E_2| \leq K^{\frac{3}{2}}$.
  - We have: $|E_2| \leq |\phi_k(E_2)| \times |\phi_{i,j}(E_2)| \leq 2K$.
- **Total:** $|E| = |E_1| + |E_2| \leq \frac{K^2}{M} + 2K$. (instead of: $|E| \leq K^{\frac{3}{2}}$)

$\Rightarrow$ When M is big, we gain a $\sqrt{K}$ factor in the asymptotic bound.

Results

- Proof automated/integrated to IOLB [Olivry et al, PLDI'20]
  Demo: `https://iocomplexity.corse.inria.fr/`

Results

- Proof automated/integrated to IOLB [Olivry et al, PLDI'20]
  Demo: `https://iocomplexity.corse.inria.fr/`

- Asymptotic I/O bounds of kernels with hourglass:

Kernel	Old bound	New bound (hourglass)	Upper bound
MGS	$\Omega\left(\frac{MN^2}{\sqrt{S}}\right)$	$\Omega\left(\frac{M^2N(N-1)}{S+M}\right)$	$O\left(\frac{M^2N^2}{S}\right)$
QR HH A2V	$\Omega\left(\frac{MN^2}{\sqrt{S}}\right)$	$\Omega\left(\frac{MN^2(N-M)}{N-M-S}\right)$	$O\left(\frac{M^2N^2}{S}\right)$
QR HH V2Q	$\Omega\left(\frac{MN^2}{\sqrt{S}}\right)$	$\Omega\left(\frac{MN^2(N-M)}{N-M-S}\right)$	$O\left(\frac{M^2N^2}{S}\right)$
GEBD2	$\Omega\left(\frac{MN^2}{\sqrt{S}}\right)$	$\Omega\left(\frac{MN^2(M-N+1)}{8(S+M-N+1)}\right)$	$O\left(MN^2\right)$
GEHD2	$\Omega\left(\frac{N^3}{\sqrt{S}}\right)$	$\Omega\left(\frac{N^4}{N+2S}\right)$	$O\left(N^3\right)$
SYTD2 (new)	$\Omega\left(\frac{N^3}{\sqrt{S}}\right)$	$\Omega\left(\frac{N^4}{N+2S-2}\right)$	$O\left(N^3\right)$

[Bonus slide] "Triangular" hourglass

- Width of hourglass might vary (with temporal dim).
  $+$ For our bound, need to use the minimum of the width.

$\Rightarrow$ Issue when this minimum is 1.

[Bonus slide] "Triangular" hourglass

- Width of hourglass might vary (with temporal dim).
  $+$ For our bound, need to use the minimum of the width.

$\Rightarrow$ Issue when this minimum is 1.

- **Solution:** loop splitting transformation.
  - Does not change the CDAG.
  - Hourglass detected on the "wide" part of the split.
  - Adjust where to split to deduce the best bound.