Computational Science on Many-Core Architectures

360.252

Josef Weinbub
Karl Rupp

Institute for Microelectronics, TU Wien
http://www.iue.tuwien.ac.at/

Zoom Channel 621 2711 2607
Wednesday, December 6, 2023

Conjugate Gradients

Pseudocode

Choose x_0
$p_0 = r_0 = b - Ax_0$
For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

BLAS-based Implementation

-
SpMV, AXPY
For $i = 0$ until convergence

1. SpMV
2. DOT
3. -
4. AXPY
5. AXPY
6. DOT
7. -
8. AXPY

EndFor

Conjugate Gradients

Pseudocode

Choose x_0
$p_0 = r_0 = b - Ax_0$
For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

BLAS-based Implementation

-
SpMV, AXPY
For $i = 0$ until convergence

1. SpMV
2. DOT $\leftarrow$ Global sync!
3. -
4. AXPY
5. AXPY
6. DOT $\leftarrow$ Global sync!
7. -
8. AXPY

EndFor

Conjugate Gradients

Pseudocode

Choose x_0
$p_0 = r_0 = b - Ax_0$
For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

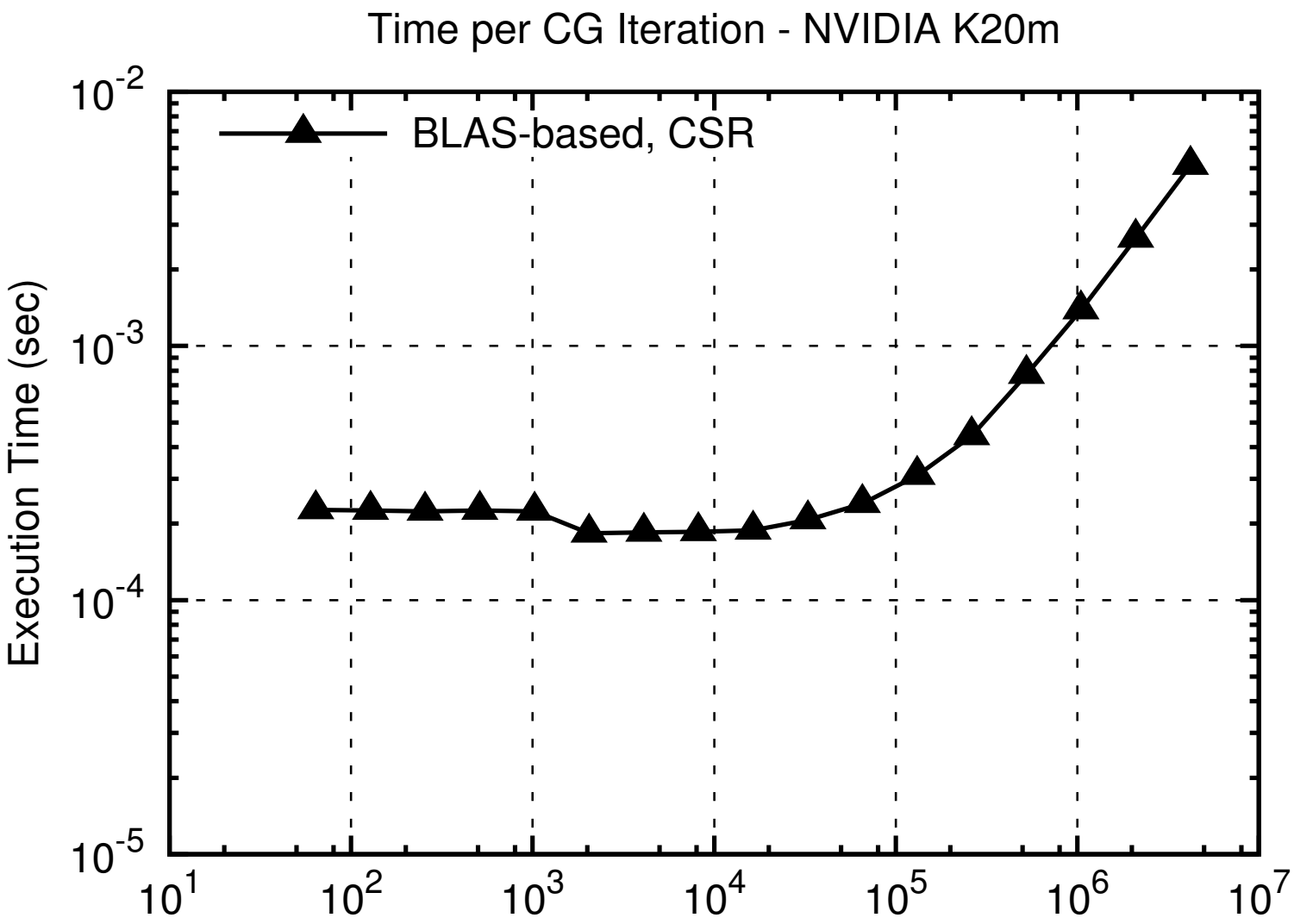
BLAS-based Implementation

-
SpMV, AXPY
For $i = 0$ until convergence

1. SpMV $\leftarrow$ No caching of Ap_i
2. DOT $\leftarrow$ Global sync!
3. -
4. AXPY
5. AXPY $\leftarrow$ No caching of r_{i+1}
6. DOT $\leftarrow$ Global sync!
7. -
8. AXPY

EndFor

Conjugate Gradients



- (2D Finite Difference Discretization)

Conjugate Gradient

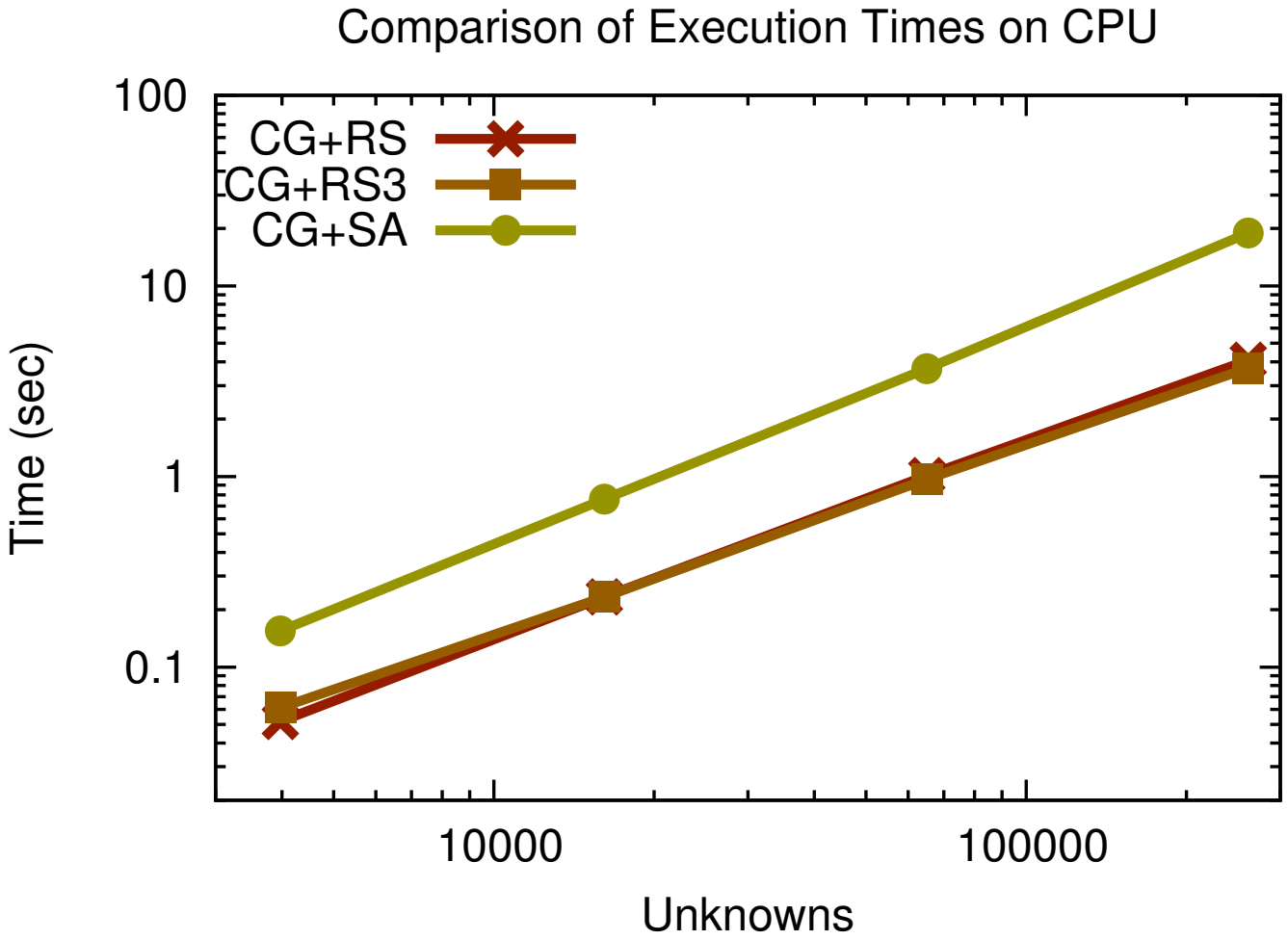
Implications

- Kernel launches expensive
- Delicate balance for preconditioners

Conjugate Gradient

Implications

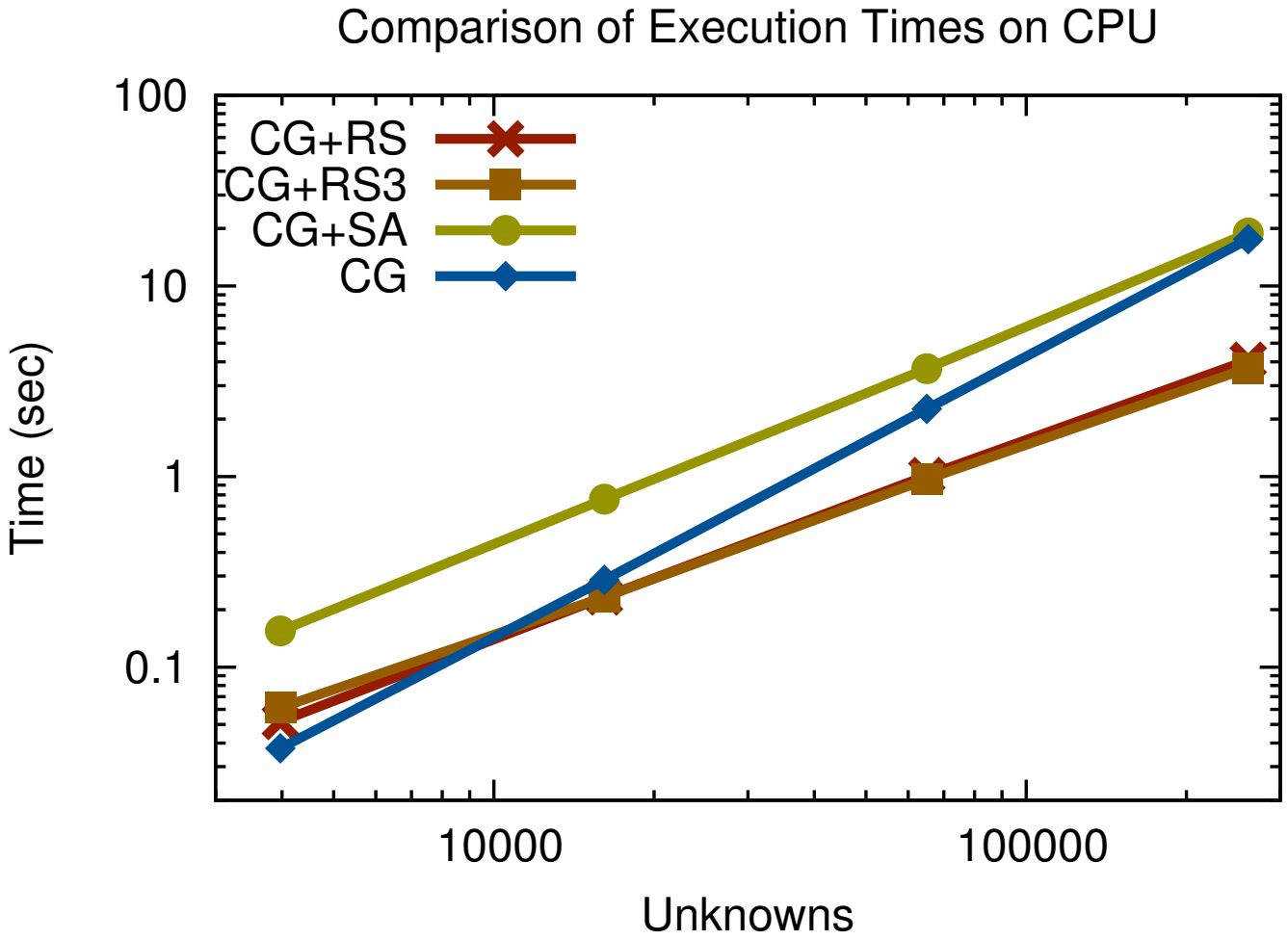
- Kernel launches expensive
- Delicate balance for preconditioners



Conjugate Gradient

Implications

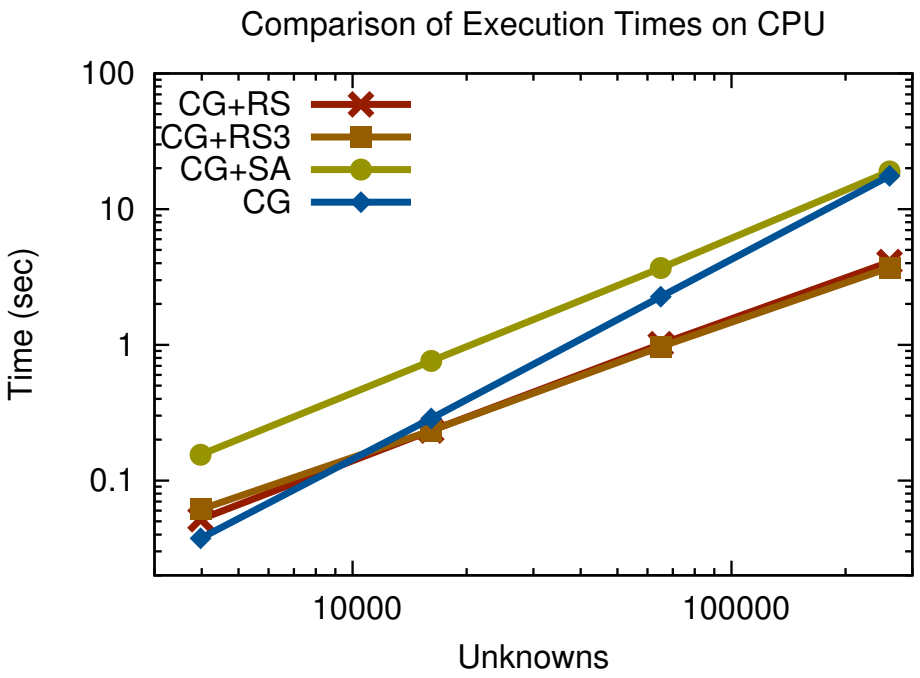
- Kernel launches expensive
- Delicate balance for preconditioners

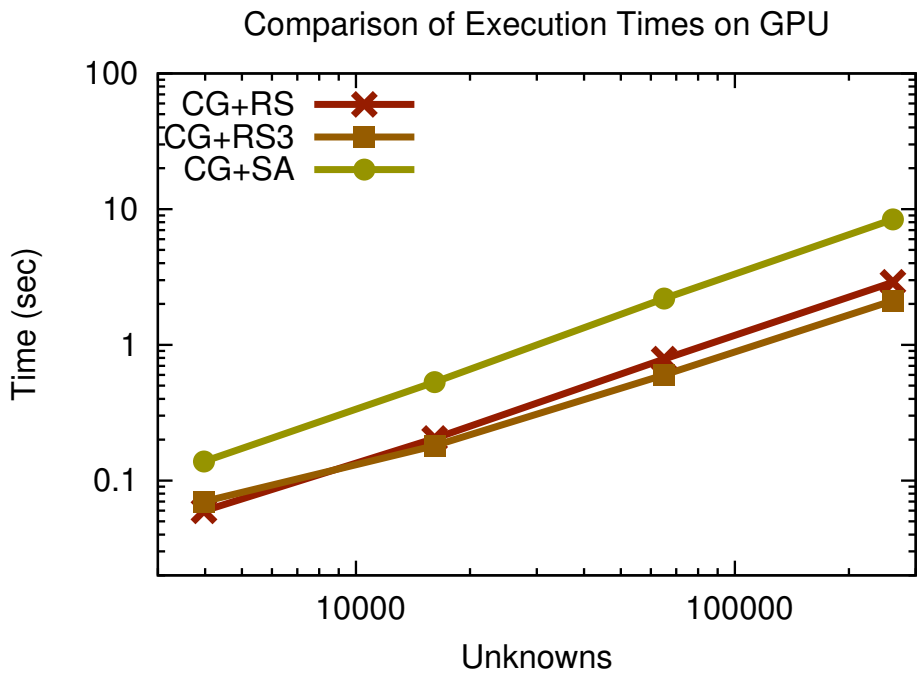


Conjugate Gradient

Implications

- Kernel launches expensive
- Delicate balance for preconditioners

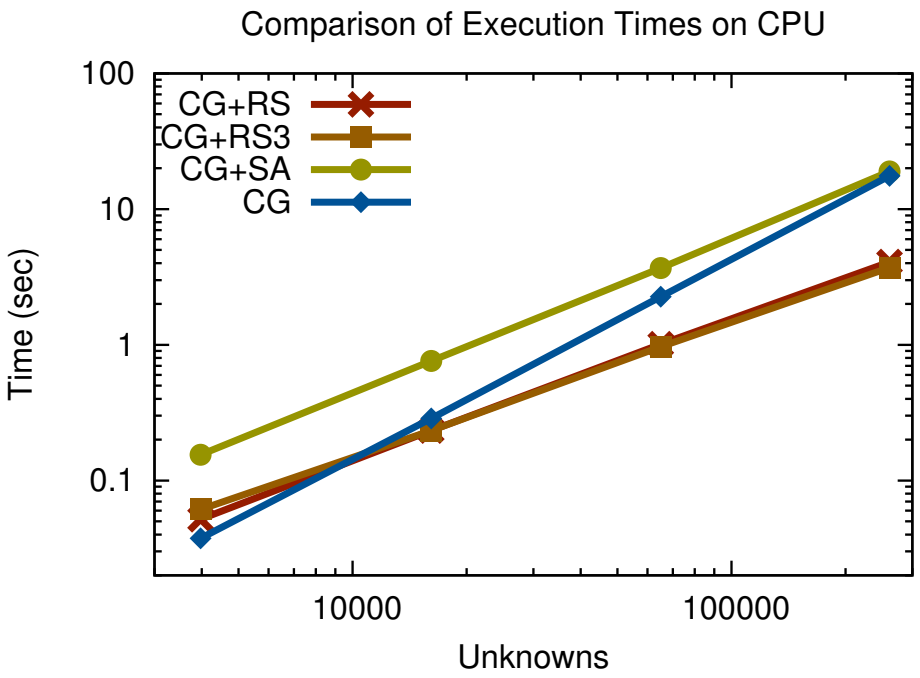


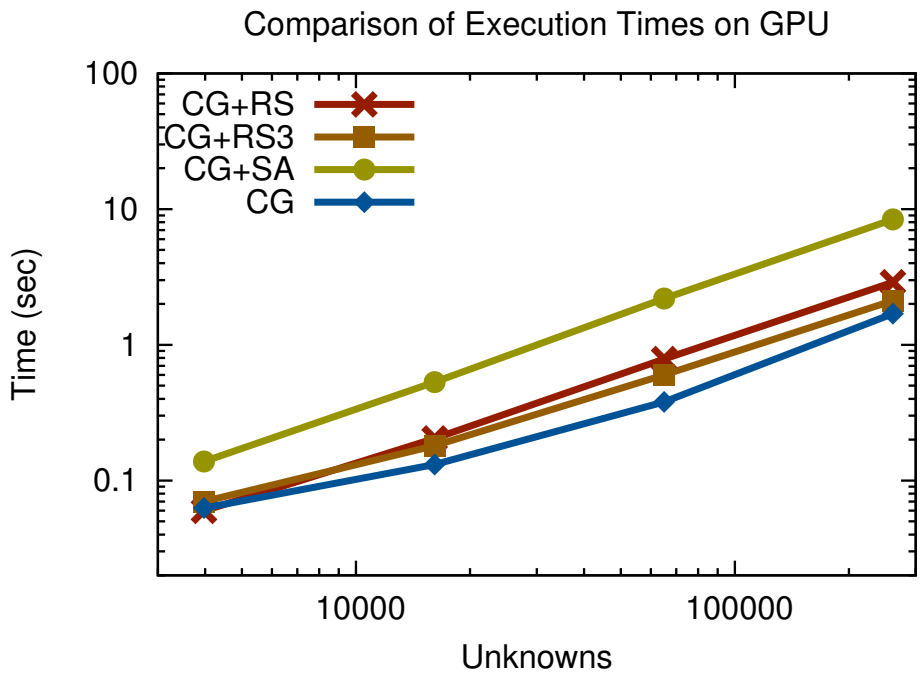


Conjugate Gradient

Implications

- Kernel launches expensive
- Delicate balance for preconditioners





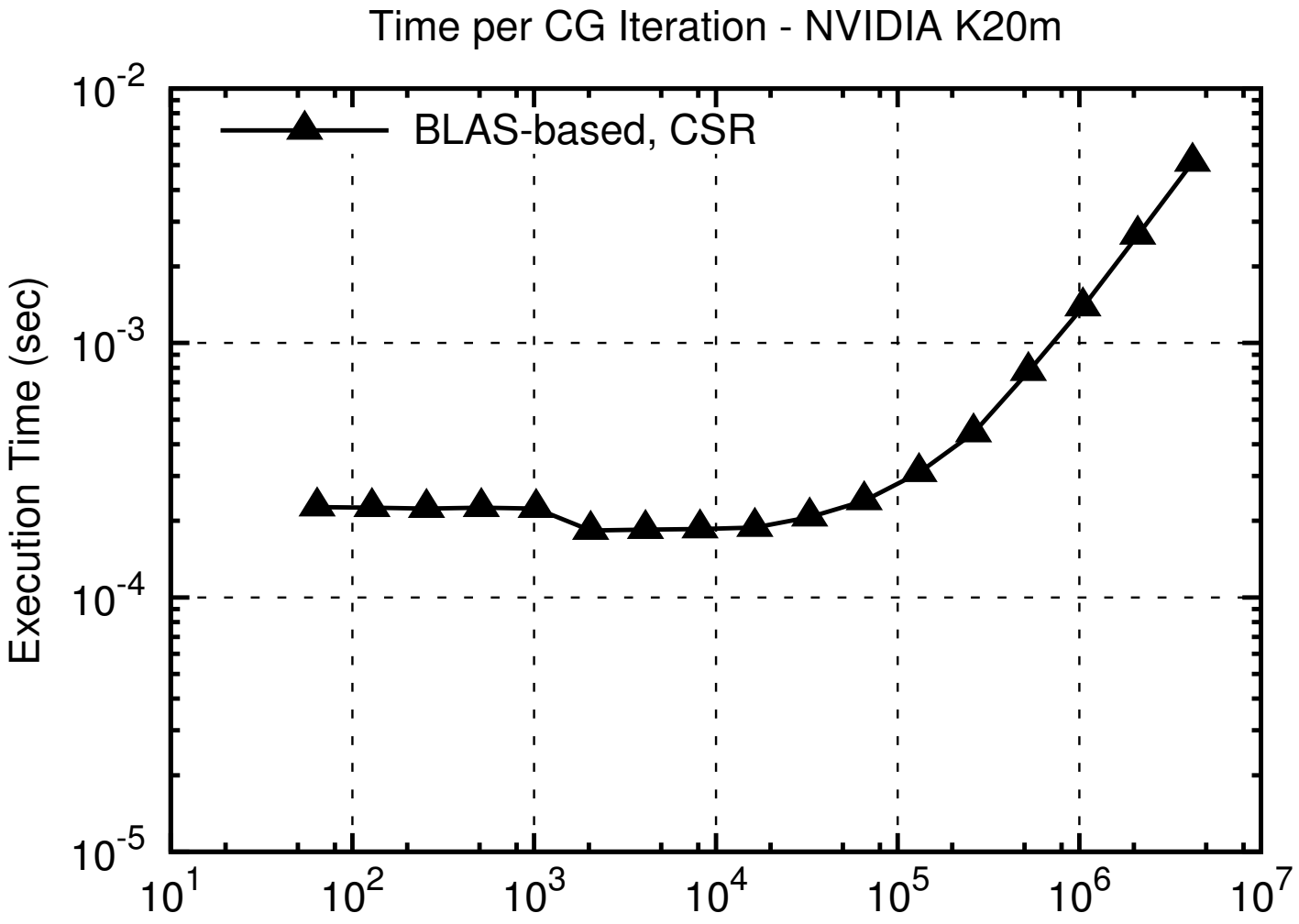
Conjugate Gradient Optimizations

Optimization 1

- Get best performance out of SpMV
- Compare different sparse matrix types

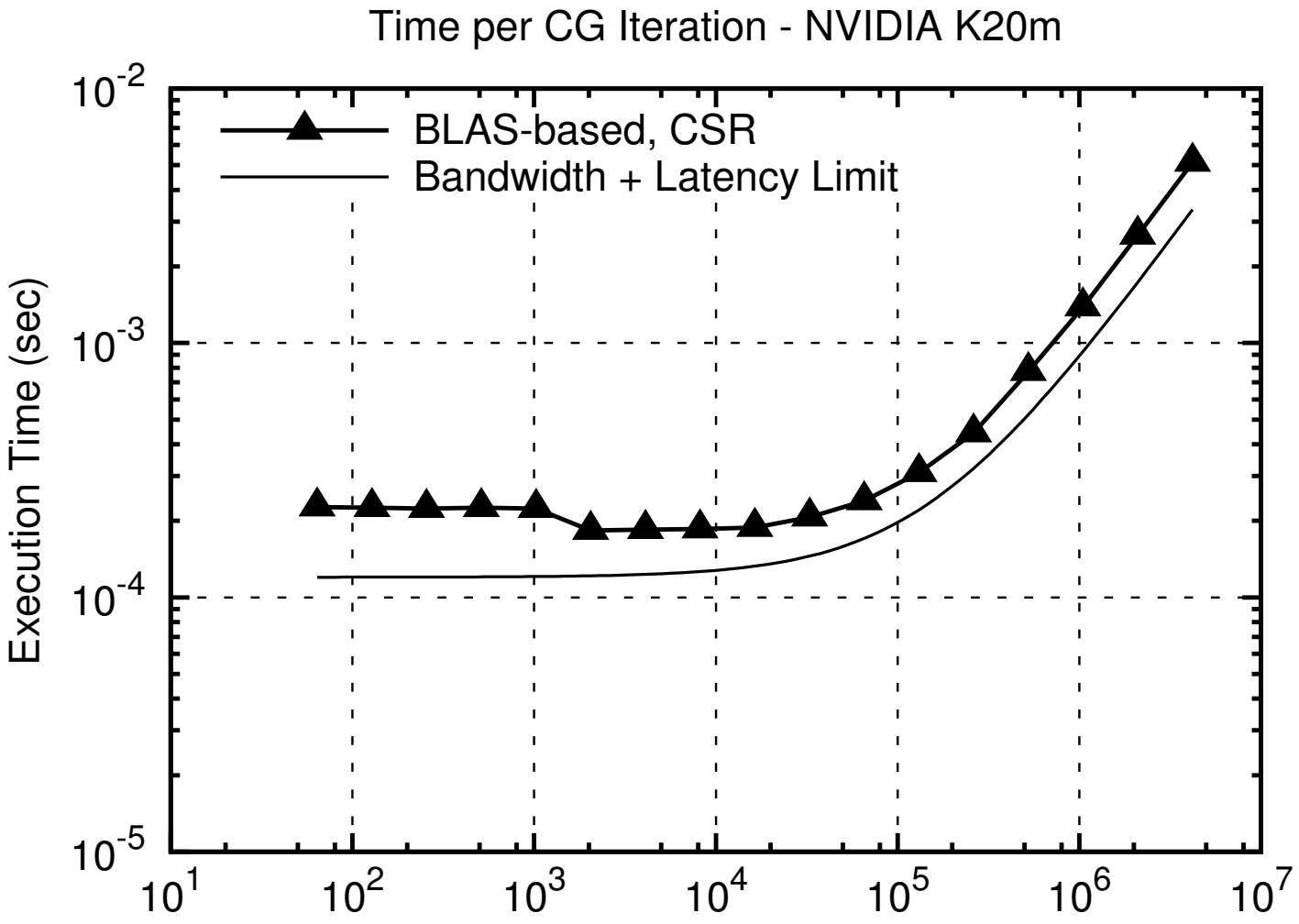
Cf.: N. Bell: Implementing sparse matrix-vector multiplication on throughput-oriented processors. *Proc. SC '09*

Conjugate Gradients



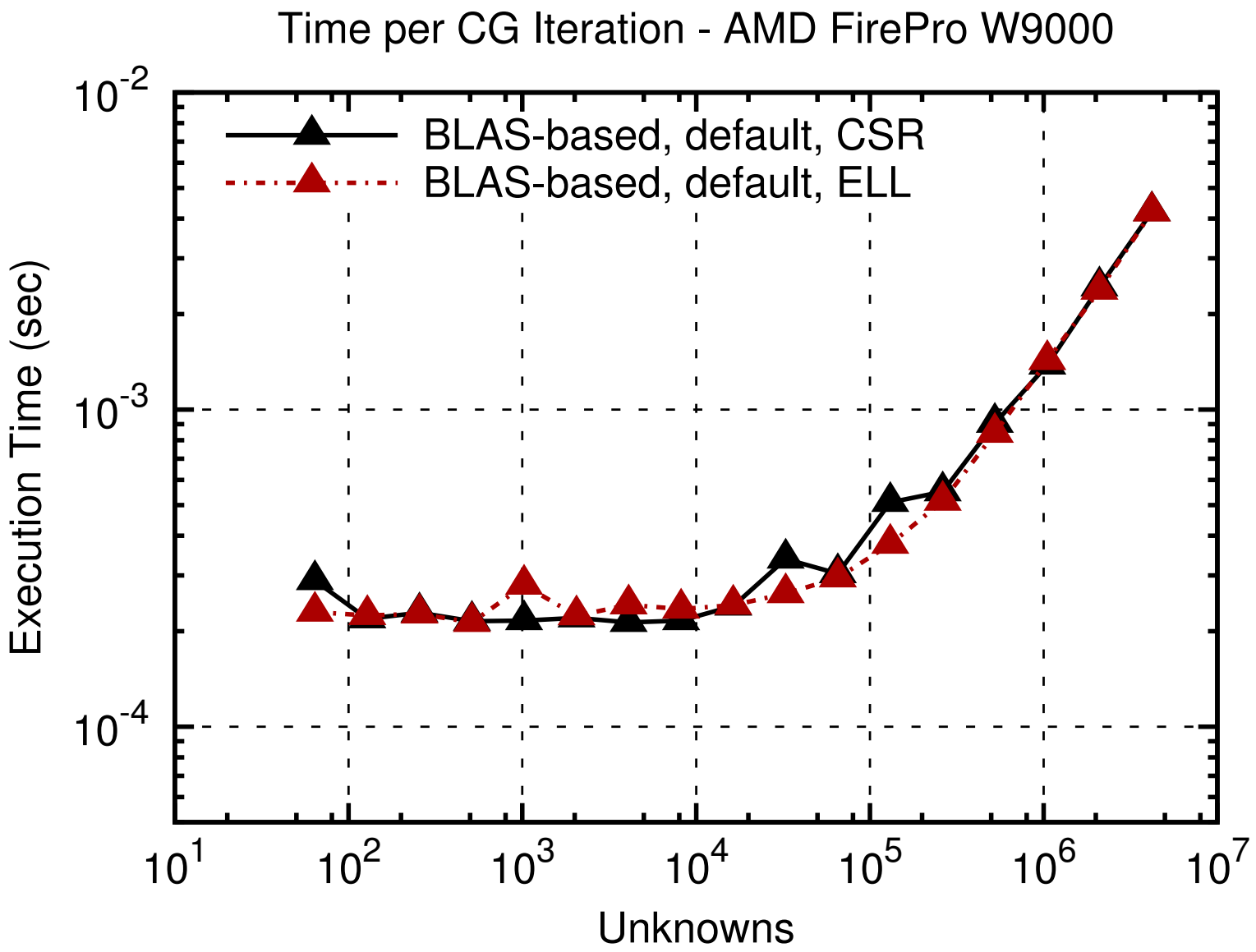
- (2D Finite Difference Discretization)

Conjugate Gradients



- (2D Finite Difference Discretization)

Conjugate Gradients



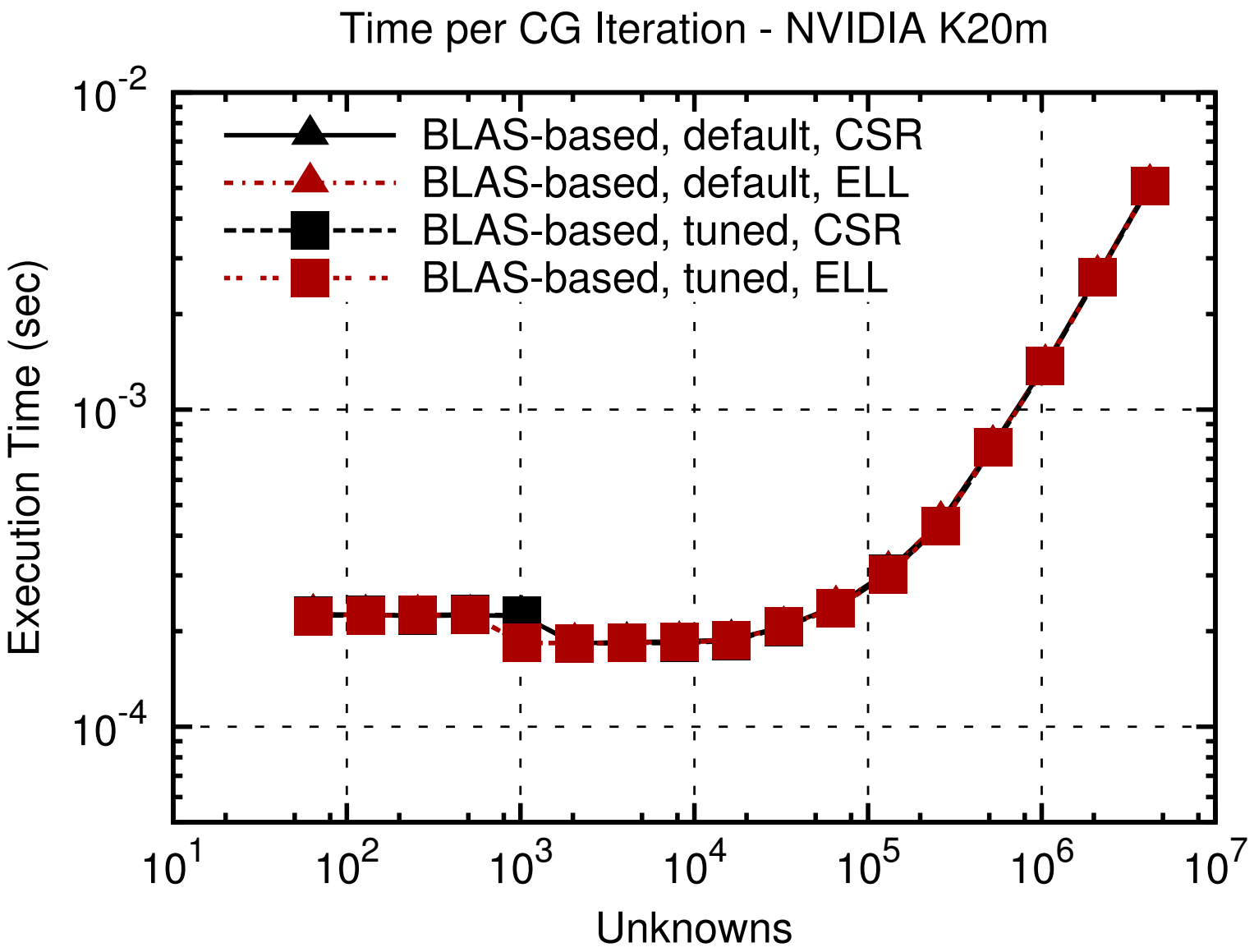
- (2D Finite Difference Discretization)

Conjugate Gradient Optimizations

Optimization 2

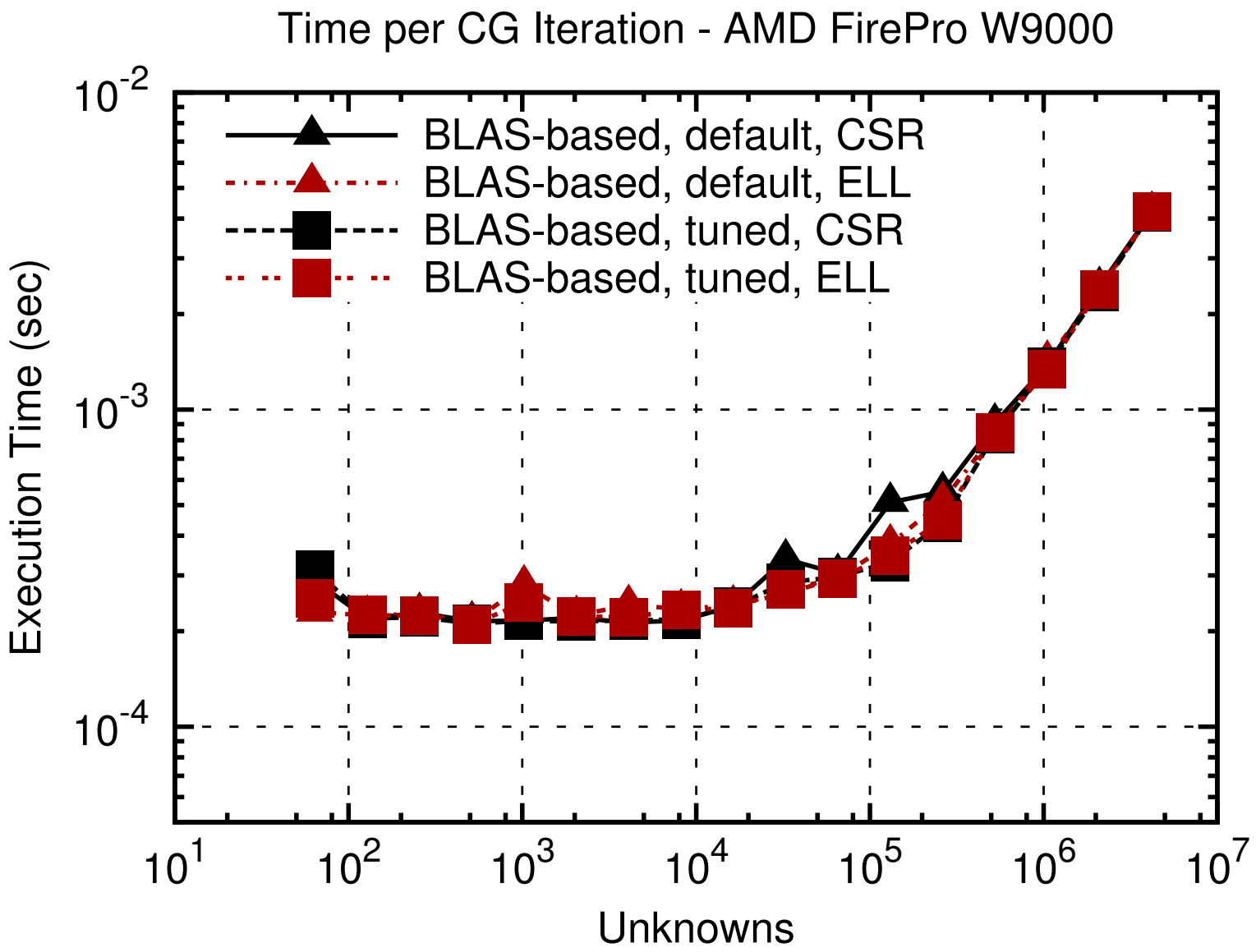
- Optimize kernel parameters for each operation

Conjugate Gradients



- (2D Finite Difference Discretization)

Conjugate Gradients



- (2D Finite Difference Discretization)

Conjugate Gradient Optimizations

Optimization 3: Rearrange the algorithm

- Remove unnecessary reads
- Remove unnecessary synchronizations
- Use custom kernels instead of standard BLAS

Conjugate Gradients

Standard CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

Conjugate Gradients

Standard CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

Pipelined CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 1$ until convergence

1. $i = 1$: Compute α_0, β_0, Ap_0
2. $x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$
3. $r_i = r_{i-1} - \alpha_{i-1} Ap_{i-1}$
4. $p_i = r_i + \beta_{i-1} p_{i-1}$
5. Compute and store Ap_i
6. Compute $\langle Ap_i, Ap_i \rangle$, $\langle p_i, Ap_i \rangle$, $\langle r_i, r_i \rangle$
7. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
8. $\beta_i = (\alpha_i^2 \langle Ap_i, Ap_i \rangle - \langle r_i, r_i \rangle) / \langle r_i, r_i \rangle$

EndFor

Conjugate Gradients

Standard CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

Pipelined CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 1$ until convergence

1. $i = 1$: Compute α_0, β_0, Ap_0
2. $x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$
3. $r_i = r_{i-1} - \alpha_{i-1} Ap_{i-1}$
4. $p_i = r_i + \beta_{i-1} p_{i-1}$
5. Compute and store Ap_i
6. Compute $\langle Ap_i, Ap_i \rangle$, $\langle p_i, Ap_i \rangle$, $\langle r_i, r_i \rangle$
7. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
8. $\beta_i = (\alpha_i^2 \langle Ap_i, Ap_i \rangle - \langle r_i, r_i \rangle) / \langle r_i, r_i \rangle$

EndFor

Conjugate Gradients

Standard CG

Choose x_0

$p_0 = r_0 = b - Ax_0$

For $i = 0$ until convergence

1. Compute and store Ap_i
2. Compute $\langle p_i, Ap_i \rangle$
3. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
4. $x_{i+1} = x_i + \alpha_i p_i$
5. $r_{i+1} = r_i - \alpha_i Ap_i$
6. Compute $\langle r_{i+1}, r_{i+1} \rangle$
7. $\beta_i = \langle r_{i+1}, r_{i+1} \rangle / \langle r_i, r_i \rangle$
8. $p_{i+1} = r_{i+1} + \beta_i p_i$

EndFor

Pipelined CG

Choose x_0

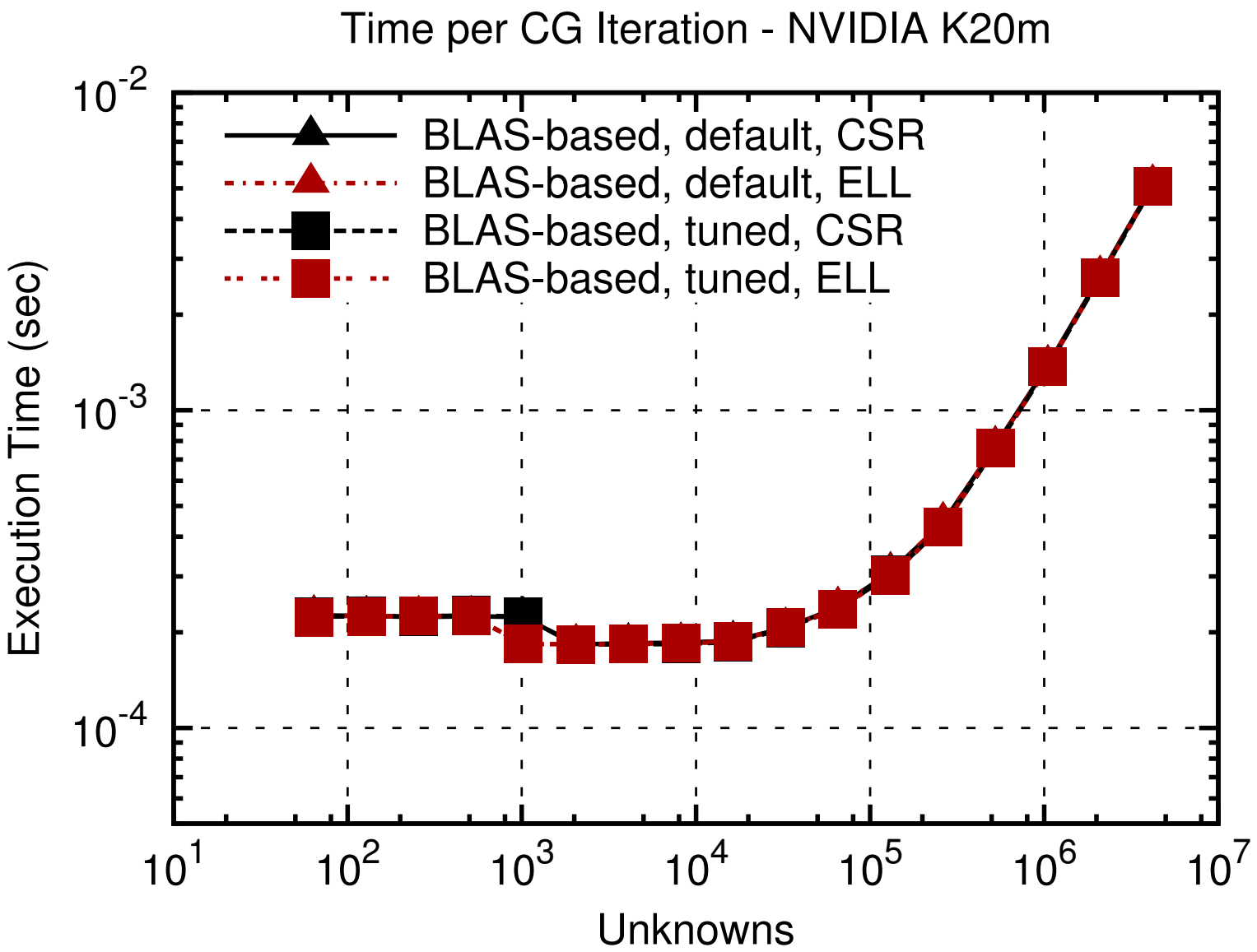
$p_0 = r_0 = b - Ax_0$

For $i = 1$ until convergence

1. $i = 1$: Compute α_0, β_0, Ap_0
2. $x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$
3. $r_i = r_{i-1} - \alpha_{i-1} Ap_{i-1}$
4. $p_i = r_i + \beta_{i-1} p_{i-1}$
5. Compute and store Ap_i
6. Compute $\langle Ap_i, Ap_i \rangle$, $\langle p_i, Ap_i \rangle$, $\langle r_i, r_i \rangle$
7. $\alpha_i = \langle r_i, r_i \rangle / \langle p_i, Ap_i \rangle$
8. $\beta_i = (\alpha_i^2 \langle Ap_i, Ap_i \rangle - \langle r_i, r_i \rangle) / \langle r_i, r_i \rangle$

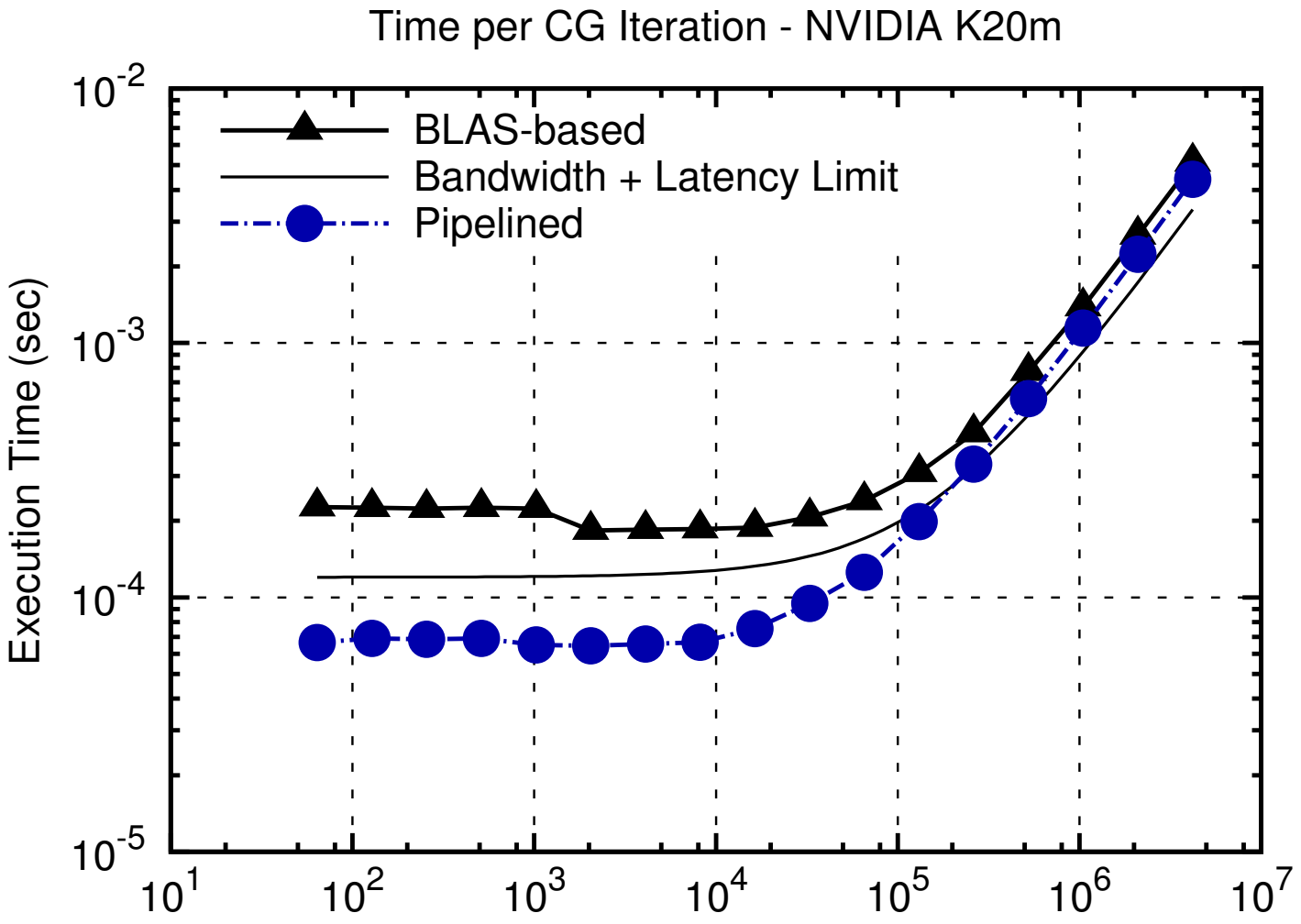
EndFor

Conjugate Gradients



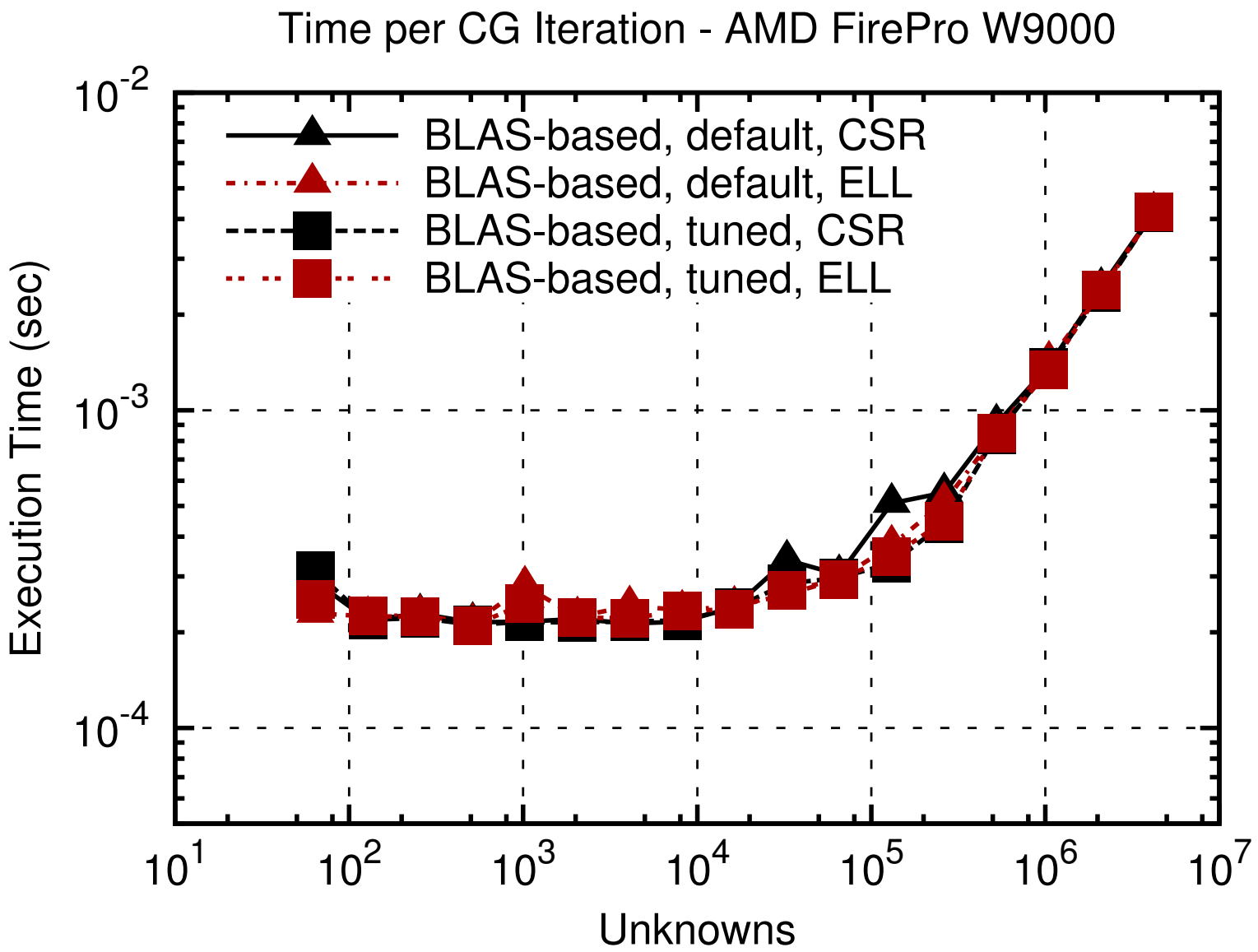
- (2D Finite Difference Discretization)

Conjugate Gradients



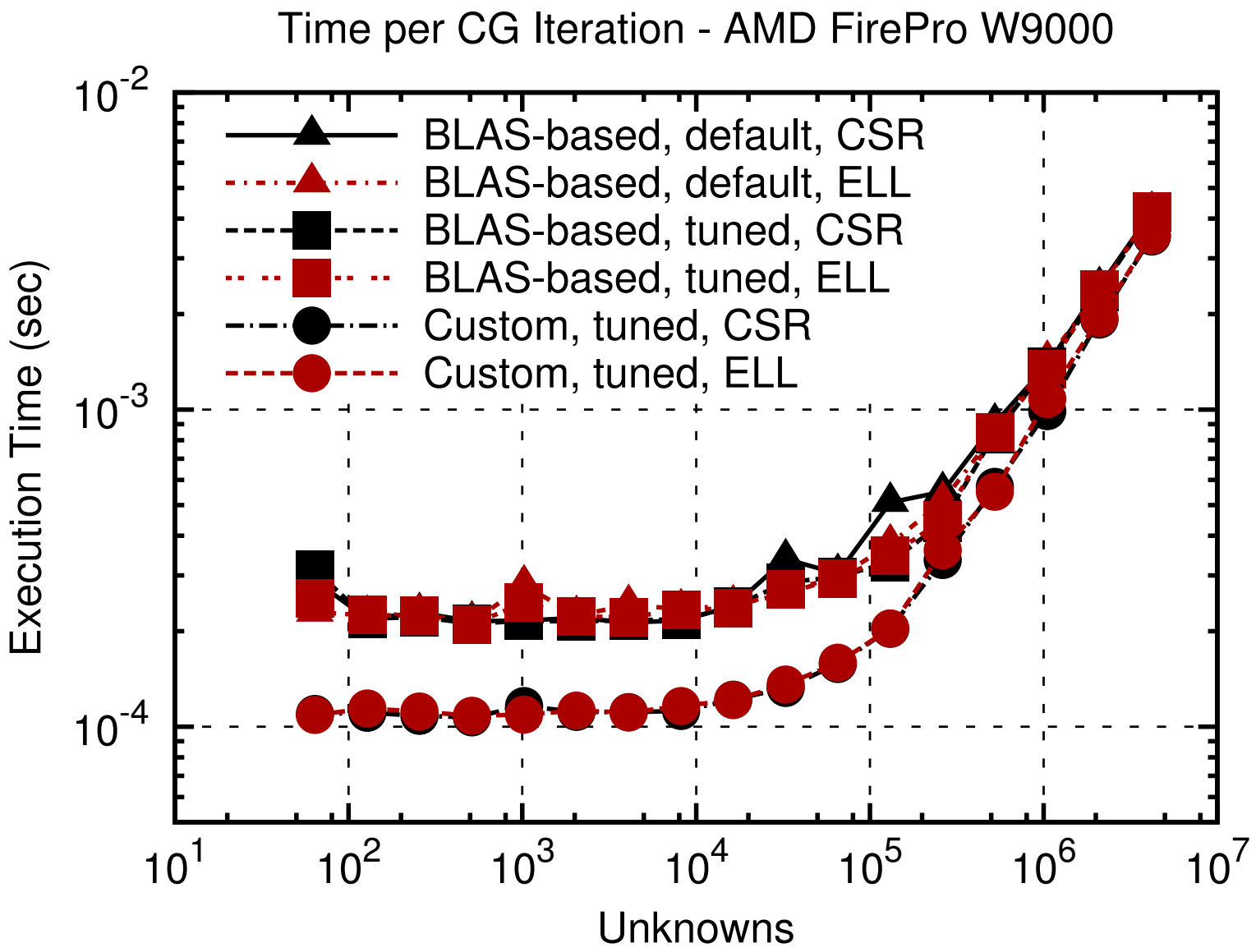
- (2D Finite Difference Discretization)

Conjugate Gradients



- (2D Finite Difference Discretization)

Conjugate Gradients



- (2D Finite Difference Discretization)