Extreme Enumeration on GPU and in Clouds

Po-Chun Kuo[1], Michael Schneider[2], Özgür Dagdelen[3], Jan Reichelt[3], Johannes Buchmann[2,3], Chen-Mou Cheng[1], and Bo-Yin Yang[4]

[1] National Taiwan University, Taipei, Taiwan
[2] Technische Universität Darmstadt, Germany
[3] Center for Advanced Security Research Darmstadt (CASED), Germany
[4] Academia Sinica, Taipei, Taiwan

CHES
Nara, Japan
2011.9.30

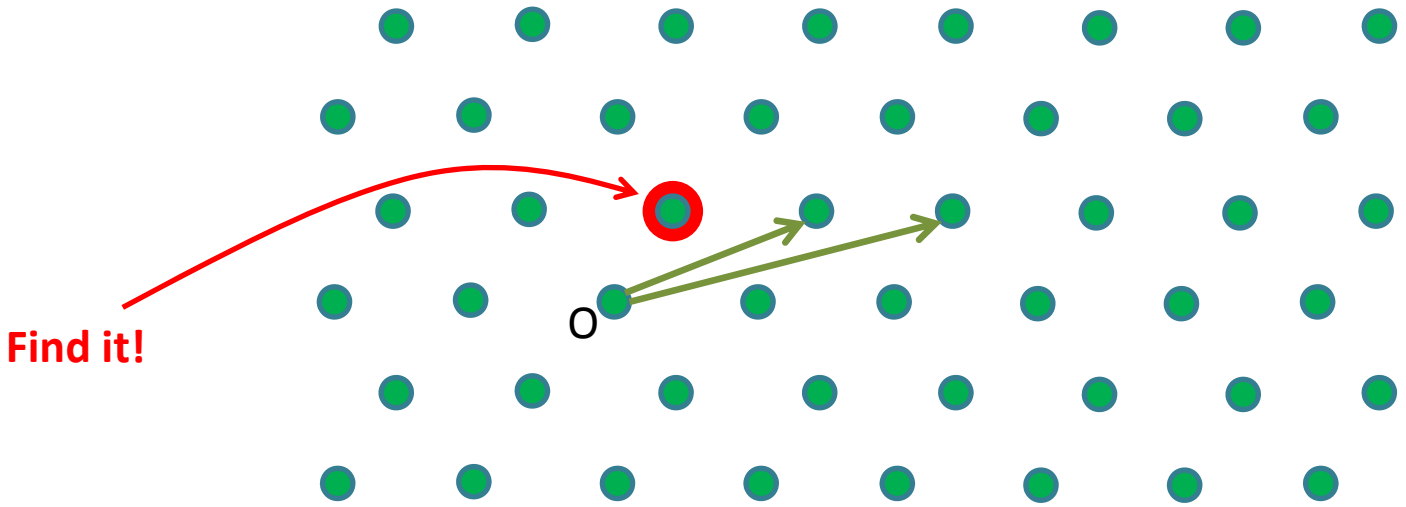
Outline

- Introduction
  - Problem Description
  - Known algorithm
  - Application
- Algorithm
- Implementation
- Results

Problem Description

- Shortest Vector Problem, SVP
- Given a basis $B \in \mathbb{R}^{n \times n}$, find shortest nonzero vector in the set of all integer combination of B

$$\|v_{min}\| = \left\| \begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \right\|$$



Problem Property

- **NP-hard**
- **Worst/average case equivalence** property
  - at 1997, Ajtai shown the equivalence between worst-case and average-case lattice problem [Ajtai'97]
- Many problems could be reduced to SVP
  - Knapsack, Subset Sum, factoring, approximate GCD problem…etc
- The **first Fully Homomorphic Encryption Scheme** is based on lattice problem is proposed at 2009 [Gentry'09]

Known Algorithm

- Approximation Algorithm
  - Lattice Basis Reduction Algorithm
  - LLL, ratio ~ 1.02^n in average case [Nguyen and Stehlé, 2006]
  - BKZ, ratio ~1.01^n in average case
    - with a parameter “block size”
  - the approximate ratio are all exponential to input dimension
- Exact Algorithm
  - Lattice Enumeration
    - Super-exponential time and polynomial space
    - But seems exponential time in practical
  - Sieve
    - both exponential time and space

Application

- Factoring polynomials over the integers or the rational numbers
  - E.g. given x^2-1, return x-1, x+1
- Finding the minimal polynomial of an algebraic number given to a good enough approximation.
  - E.g, given 1.618033, return $x^2-x+1=0$
- Factoring number with known some bit
- Approximate GCD problem
- Integer Programming
- Knapsack problem
- Subset Sum problem

Our Contributions

- **Parallelize and implement** GNR'10 extreme pruning lattice reduction on GPU and in Cloud
  - Highly parallelize, about 90% parallel benefit
  - One GTX 480 is about 12 times faster than one i7 core.
  - Extend the implementation to multiple GPUs and run on Amazon's EC2 cloud services.
- **Extend the pruning idea to pre-computing and flexible bounding function**
  - Get the same output quality and speed up more than 10x in pre-computing
- **New security level measure**
  - Tradition: Dollar-day (e.g. buy 10 machines and run 5 days)
  - New: Dollar (e.g. rent computation power from Amazon)
- **Estimate the cost of solving ASVP instances of SVP Challenge in higher dimensions**

Algorithm

- **Lattice Enumeration**
- Lattice Enumeration using Extreme Pruning

Algorithm

- Lattice Enumeration
  - Exhaustive search all the possible solutions

$$\|v_{min}\| = \left\| \begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \right\|$$

Gram-Schmidt Process

$$= \left\| \begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} \mu_{1,1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \mu_{n,1} & \cdots & \mu_{n,n} \end{bmatrix} \begin{bmatrix} b_1^* \\ \vdots \\ b_n^* \end{bmatrix} \right\|$$

$$= \left\| \begin{bmatrix} b_1^* & \cdots & b_n^* \end{bmatrix} \begin{bmatrix} c_1 + c_2\mu_{2,1} + \cdots + c_n\mu_{n,1} \\ c_2 + \cdots + c_n\mu_{n,2} \\ \vdots \\ c_n \end{bmatrix} \right\|$$

$$= \left\| \begin{bmatrix} b_n^* & \cdots & b_1^* \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} + c_n\mu_{n,n-1} \\ \vdots \\ c_2 + \cdots + c_n\,\mu_{n,2} \\ c_1 + c_2\mu_{2,1} + \cdots + c_n\mu_{n,1} \end{bmatrix} \right\|$$

- Build the search tree according to the "coefficient vector"
- Guess *bound* = $||b_1||$ or Gauss prediction
- Run **DFS** to find the shortest vector

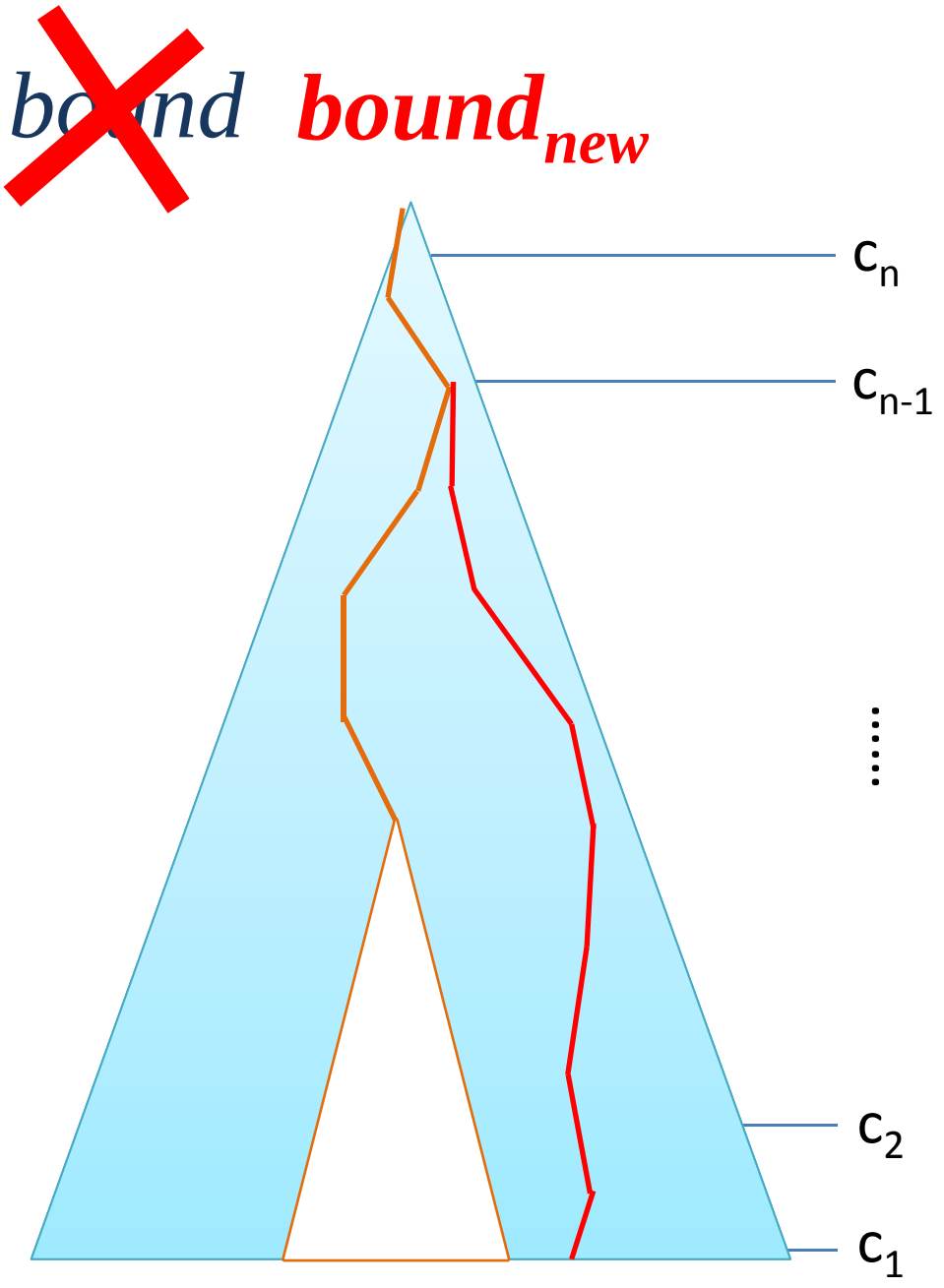
$$\|v_{min}\| = \left\| [b_n^* \quad \dots \quad b_1^*] \begin{bmatrix} c_n \\ c_{n-1} + c_n\mu_{n,n-1} \\ \vdots \\ c_2 + \cdots + c_n\mu_{n,2} \\ c_1 + c_2\mu_{2,1} + \cdots + c_n\mu_{n,1} \end{bmatrix} \right\|$$

$$|c_n| \le \sqrt{bound / \|b_n^*\|}$$

$$|c_i| \le \sqrt{\frac{bound - {}_{current}Length_i}{\|b_i^*\|}}$$

$$_{current}Length_i = \sum_{k=i+1}^{n} (x_k + \sum_{j=k+1}^{n} \mu_{j,k} x_j) \|b_k^*\|^2$$

- Guess bound
  - Early abort
  - Renew bound
  - Gauss prediction
- Extreme pruning[GNR'10]
- Parallel



Algorithm

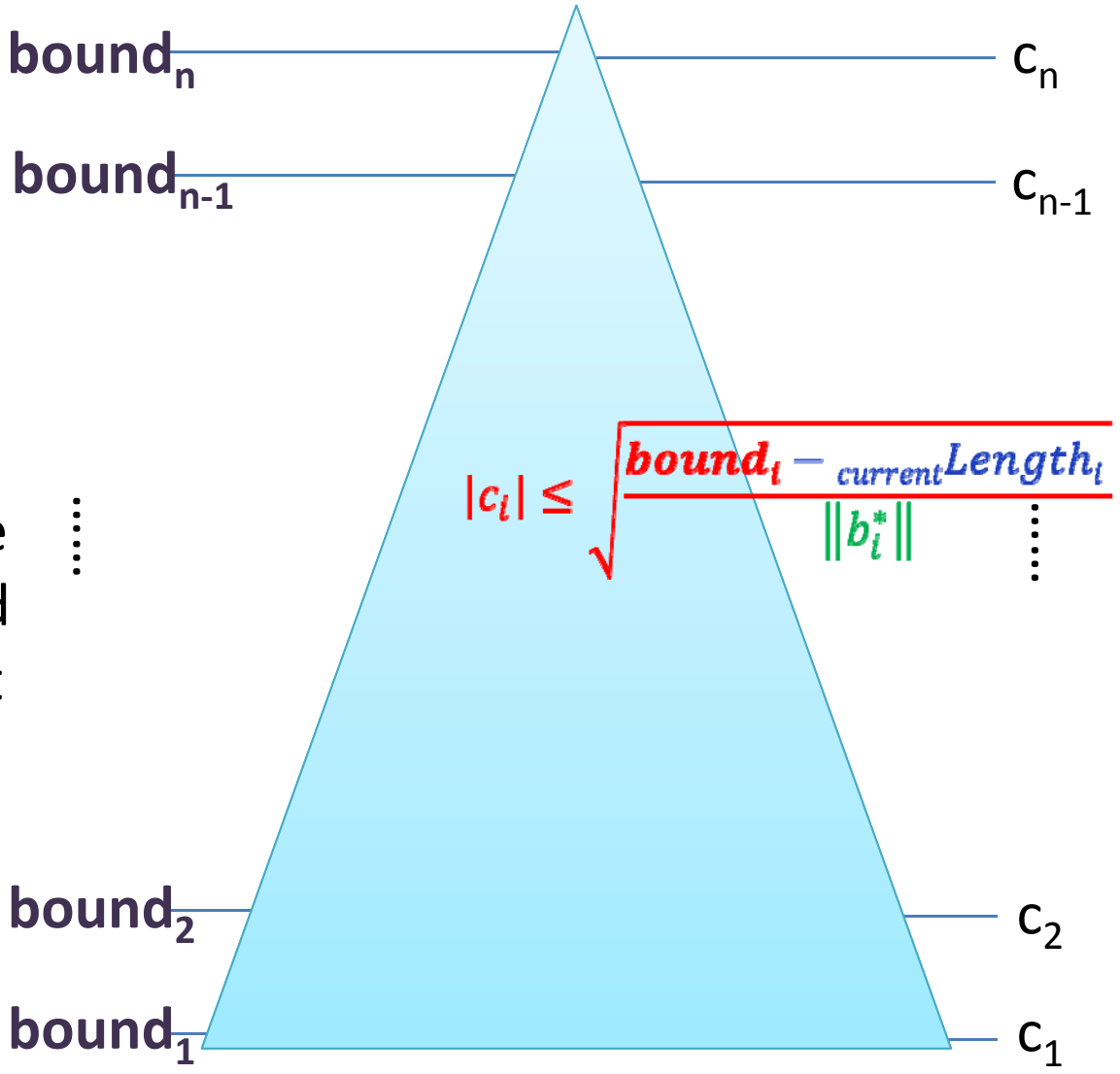
- Lattice Enumeration
- **Lattice Enumeration using Extreme Pruning**

Idea of Extreme Pruning

- Proposed by Nicolas Gama, Phong Q. Nguyen and Oded Regev
- Goal: **Maximize the reward per operation**
- Only search the space where is the most possible to find out solution
- If failed, randomized the basis and search again.

Lattice Enumeration using Extreme Pruning

- Guess **very tight bound** for each level
- If failed, do a randomize to the basis, then do Enumeration again.
- E.g.
  time for search each tree be 1000 times faster, and probability of finding out the vector be 10%,
  => gain ~100x faster



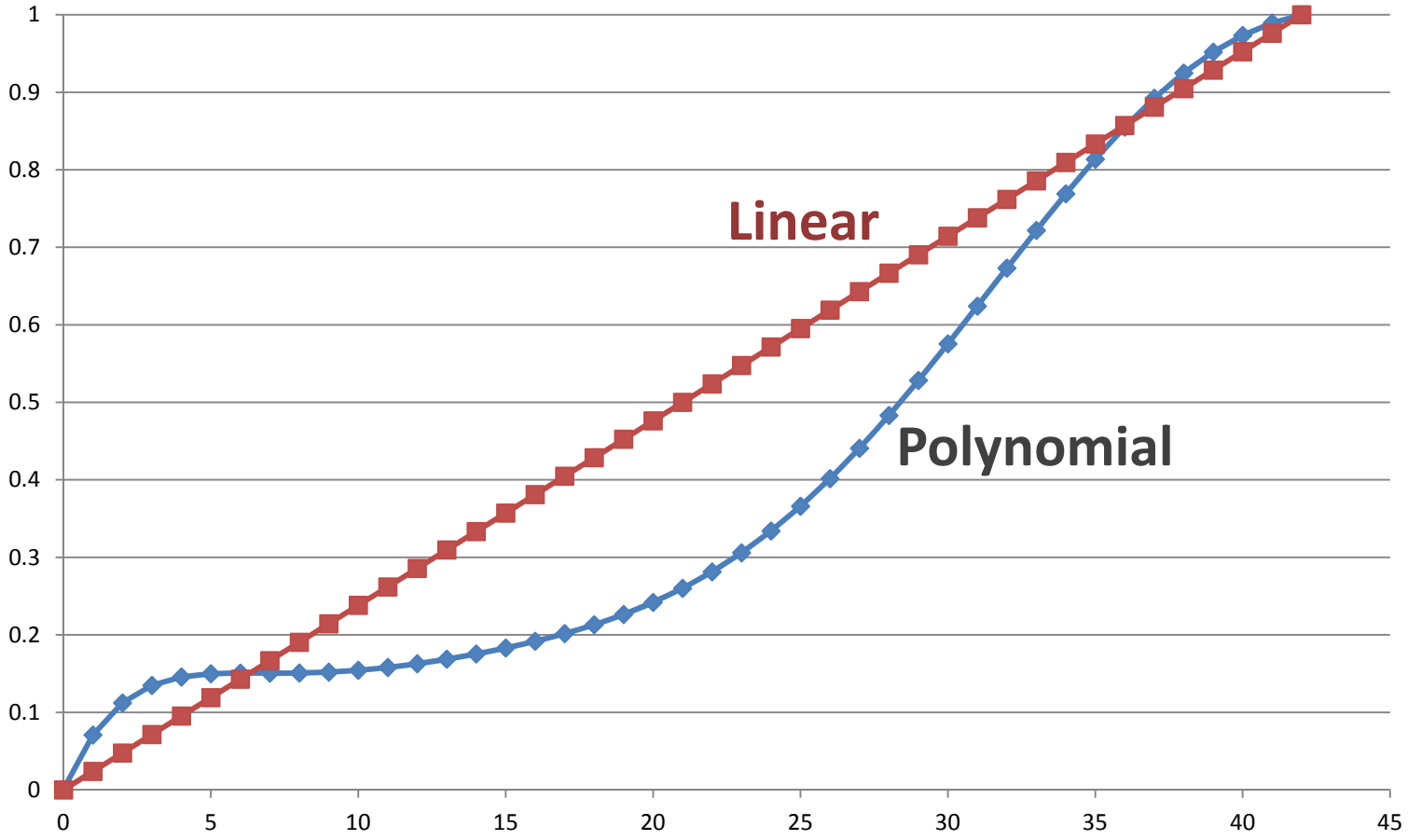
How to decide bound for each level ?

- Bounding vector $(R_1, R_2, \ldots, R_n) \in [0,1]^n$, with $R_1 \leqq R_2 \leqq \ldots \leqq R_n$
- Bounding function for level i is $R_i \cdot Guess\ Bound$

- Linear bounding function
  - $R_i = i / n$
  - Theoretical analysis -> the success probability $1/n$ [GNR'10]
  - In Practice -> the success probability is much higher
  - But the search space is still too large for high dimension…

- Polynomial: fitting the numerically optimized in [GNR'10]
  - No theoretical guarantee on the probability
  - But it performs well in practice
  - success probability is about 10% by experiment

Bounding function

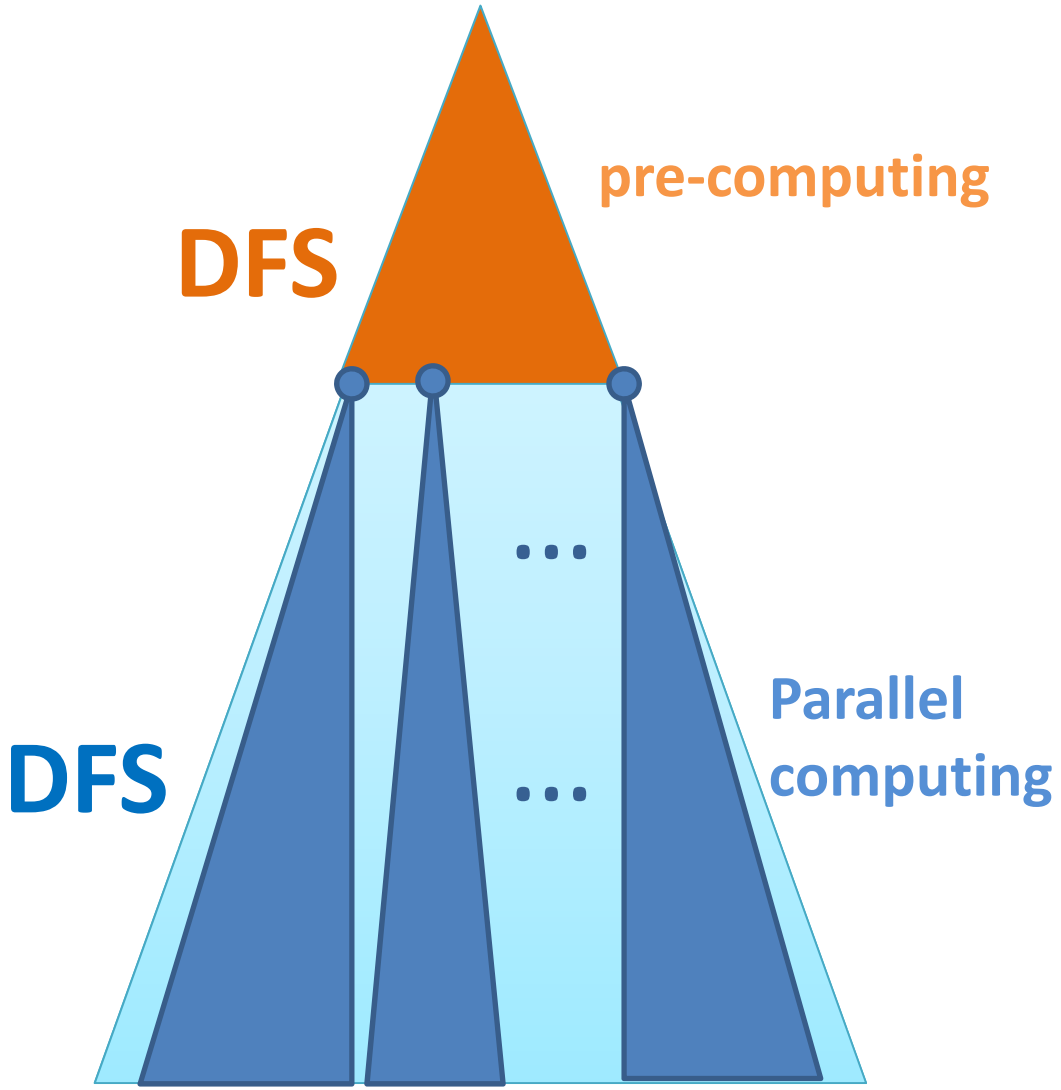


Parallel

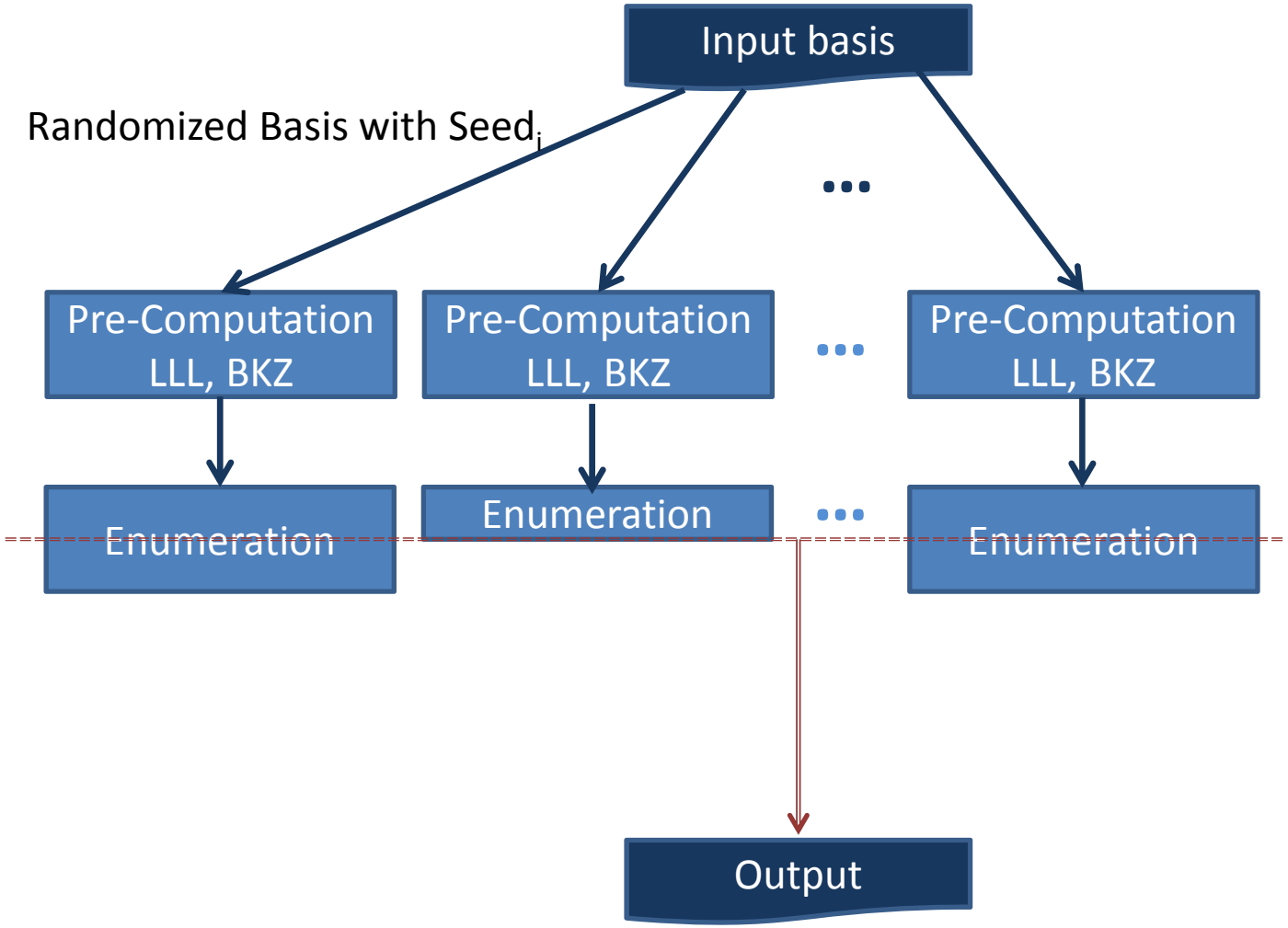
- Parallel for Enumeration
  - Parallel for ***one*** search tree
- Parallel for Extreme Enumeration
  - Parallel for ***many*** search trees

Parallel for Enumeration

- Parallel
  - $DFS_{upper\ tree}$ + $DFS_{lower\ tree}$
- $DFS_{upper\ tree}$
  - one computer
  - pre-computing
- $DFS_{lower\ tree}$
  - Parallel part
- Output the shortest vector
  - collecting the sub-tree results.



Parallel for Extreme Enumeration

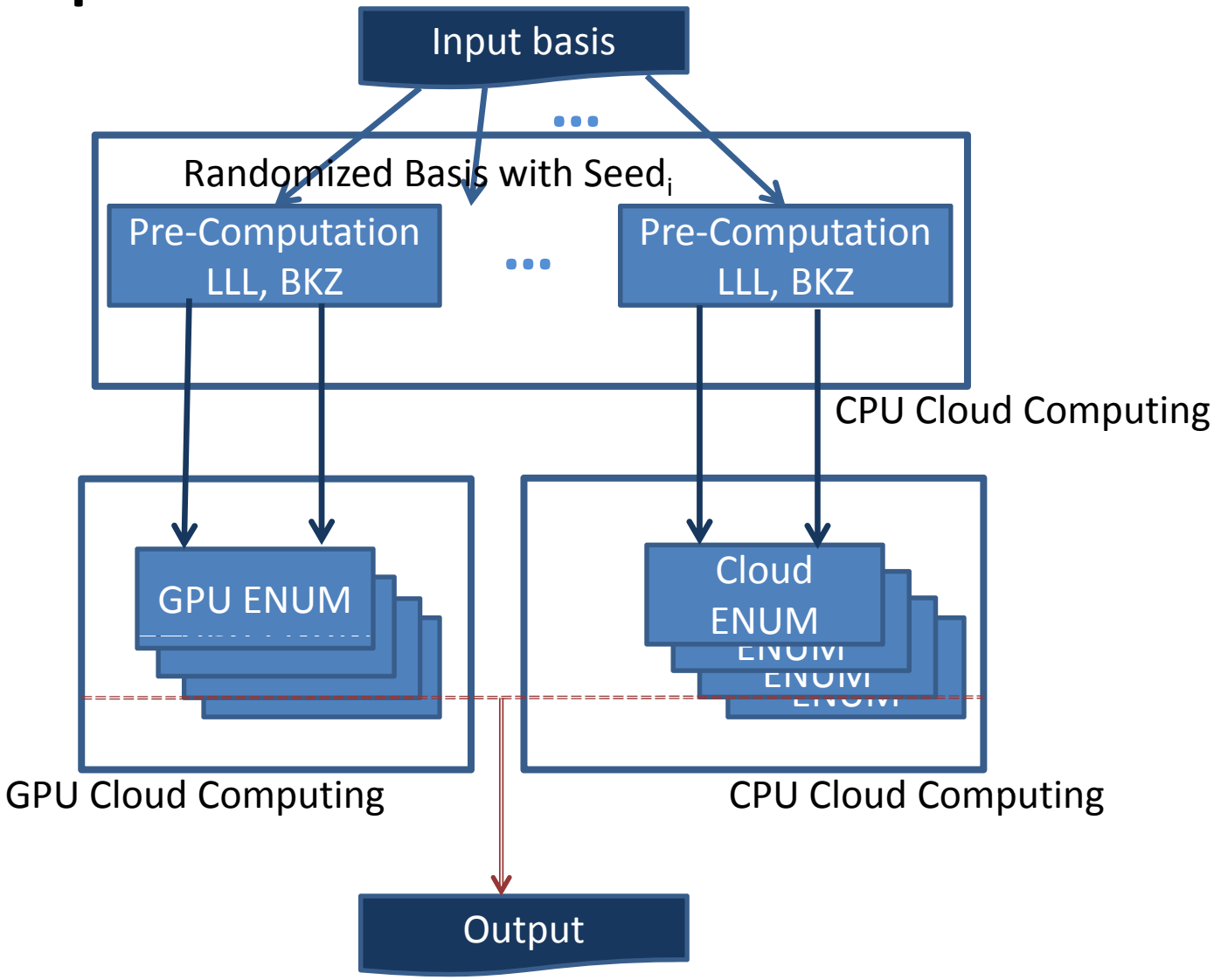


Implementation

- We have Cloud and GPU version implementations.
- Cloud version: C++ and Hadoop Streaming
- GPU version : CUDA

- Memory Used:
  - For dimension n, needs $8n^2+40n+164$ bytes
  - n=100, needs 84 KB
  - n=800, needs 5.1 MB

Implementation overview

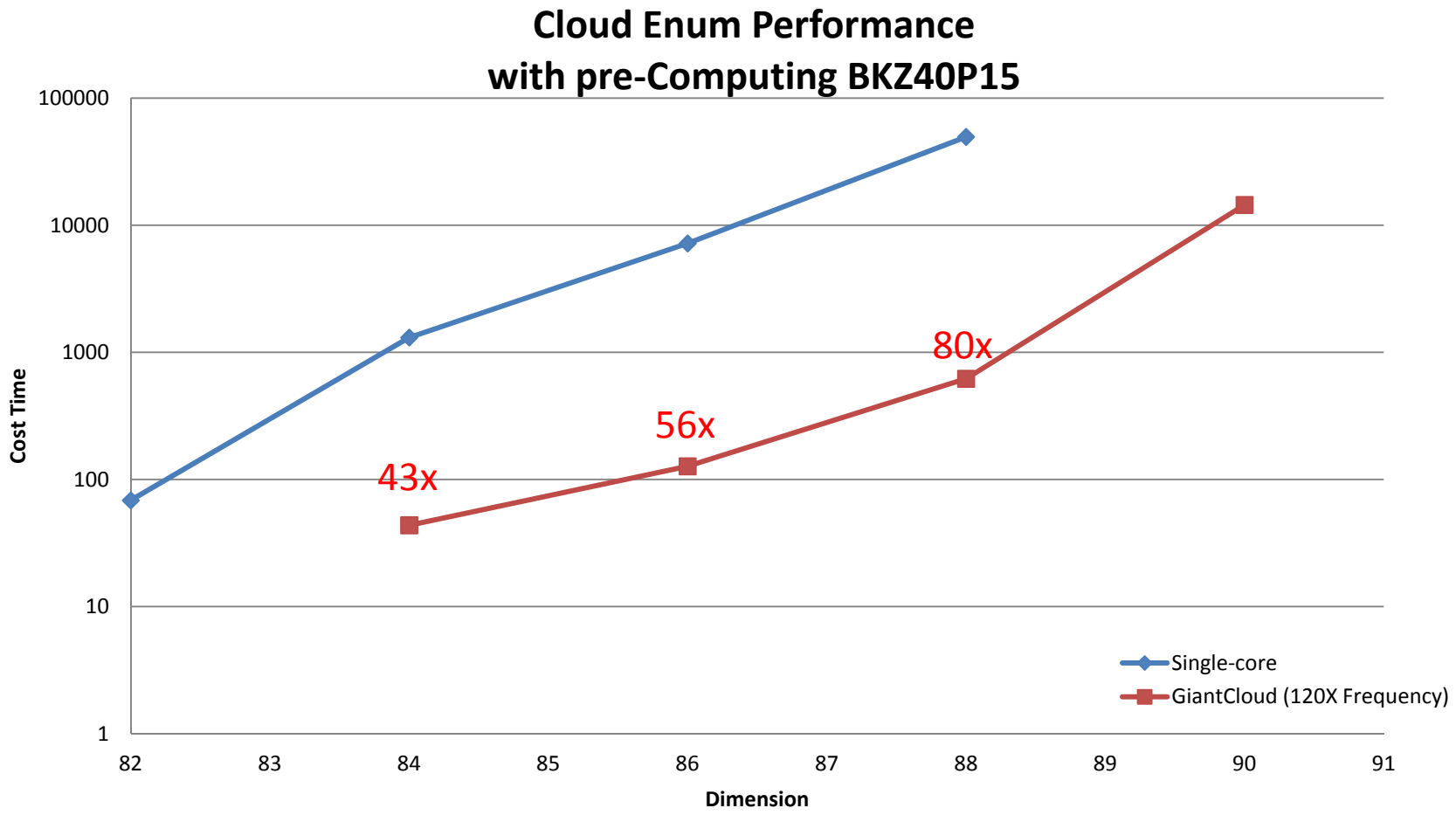


– One GTX 480 is about 12 times faster than one i7 core.

Results

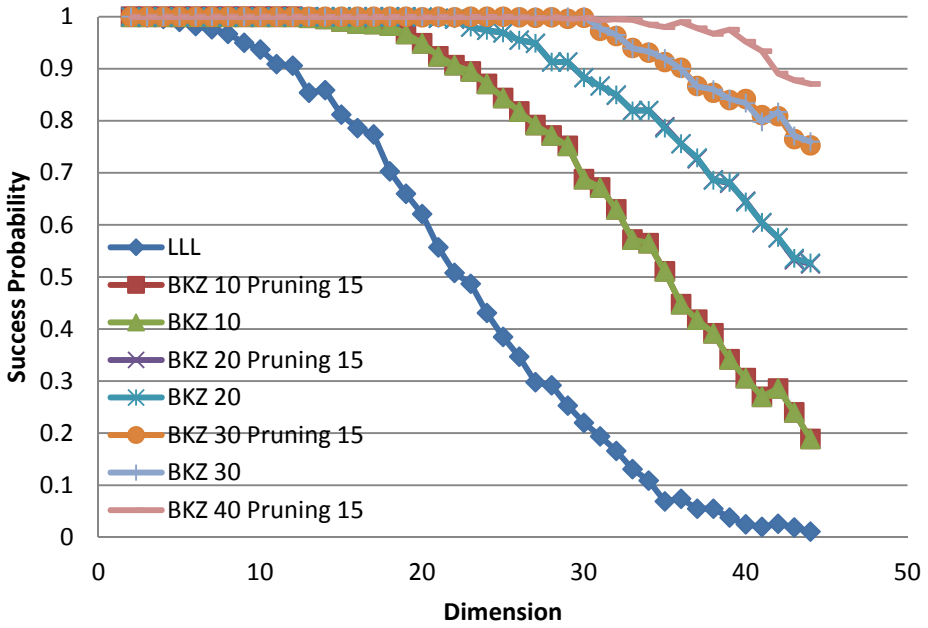
- We launch our CPU Cloud in IIS, Taiwan
  - Machine 0-1 Intel Xeon E5430
    - 2.66 (GHz) * 2(cpus) * 4(cores) * 1(thread)
  - Machine 2-3 Intel Xeon E5520
    - 2.27 (GHz) * 2(cpus) * 4(cores) * 2(threads)
  - Machine 4-8 Intel Xeon E5620
    - 2.40 (GHz) * 2(cpus) * 4(cores) * 2(threads)
  - **Total:**
    - **9 nodes, 72 physical cores, 128 virtual cores,**
    - **306 GHz**

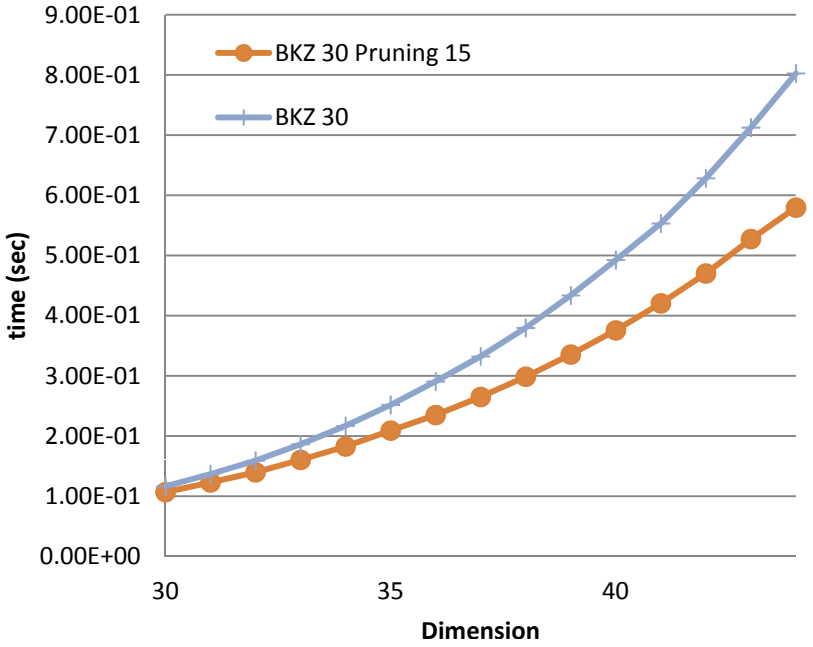
Performance compare to single-core



- Speed-up ratio is increase

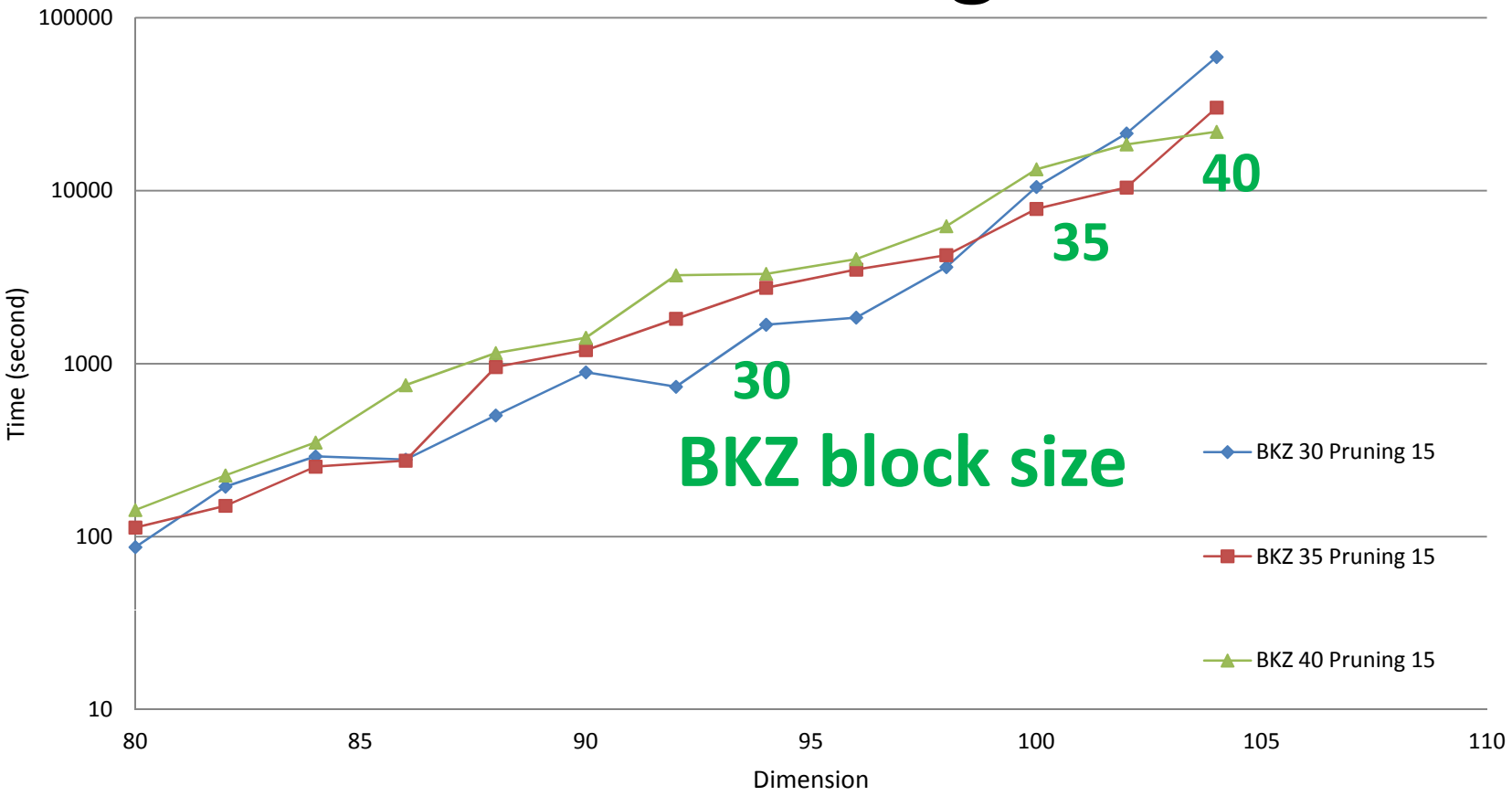
Prune BKZ in Pre-Computing





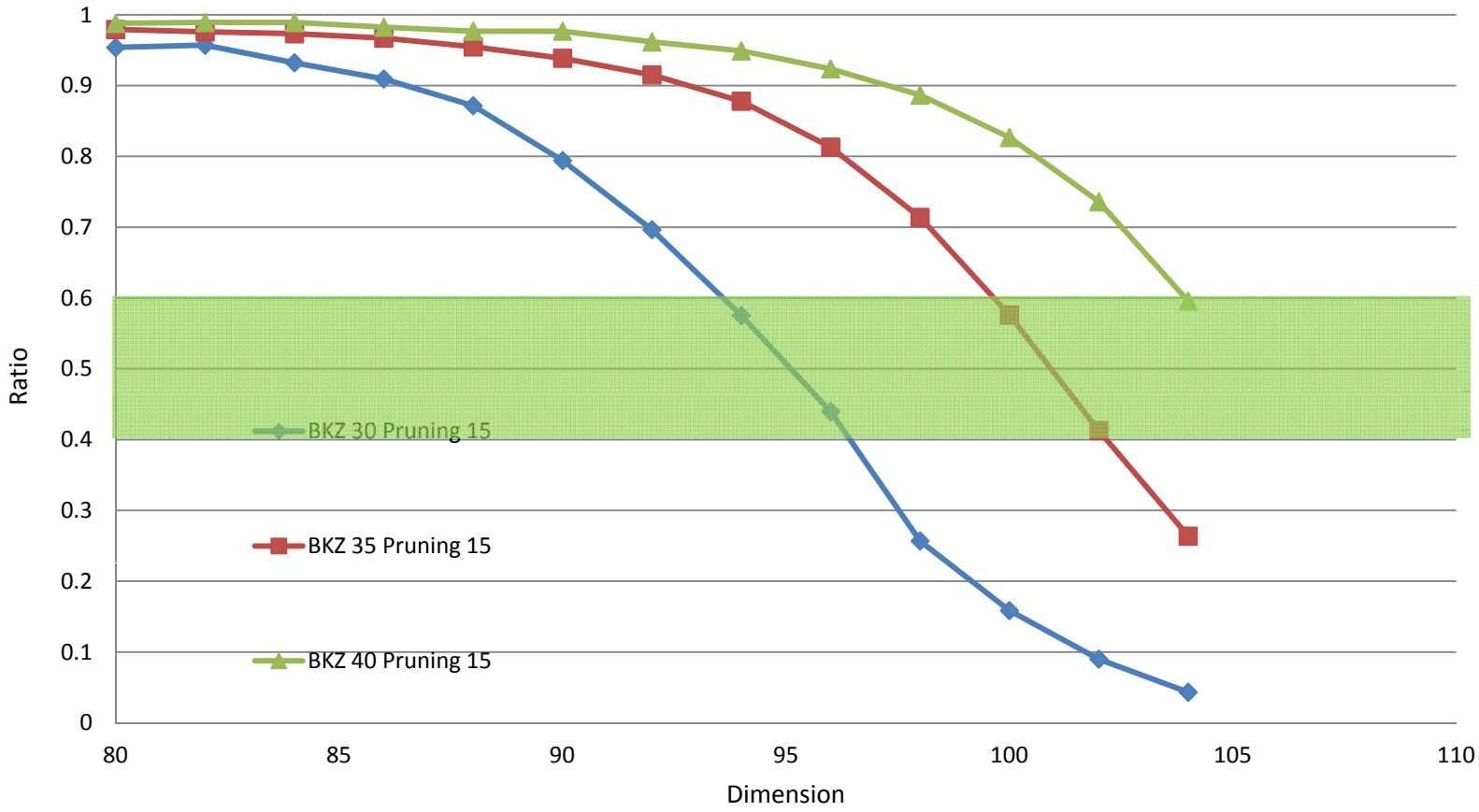
- Almost the same quality as none-pruning version
- But Cost time is much less
  - in dimension 80 -120, speed-up 10x faster

Total running time

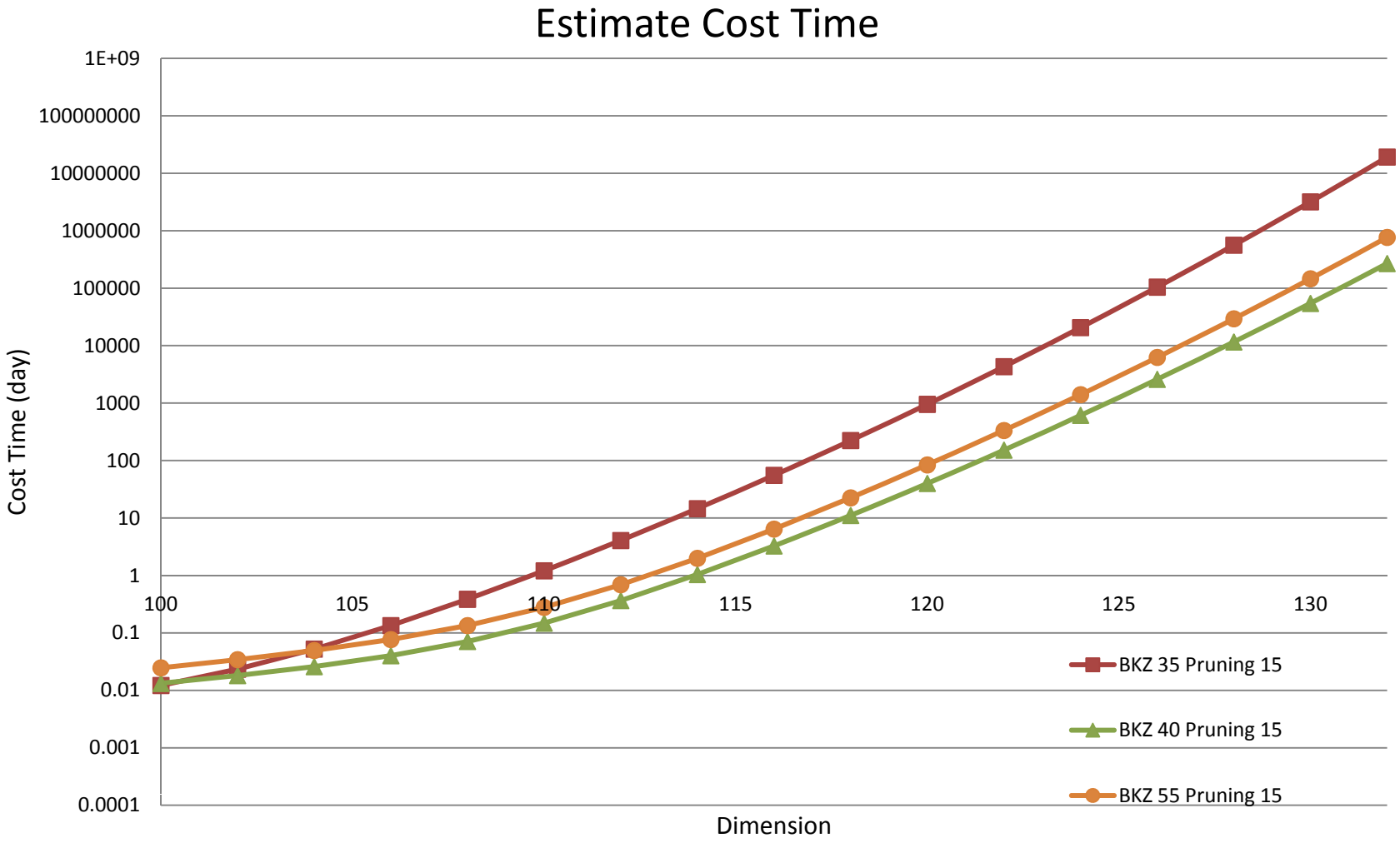


- Seems exponential increase
- Tradeoff on the pre-computing (basis quality) and extreme enumeration

BKZ / (BKZ+ ENUM)



- To combine with “total running time”, when percentage BKZ of total time is about 40-60%, it gets the min total running time



- This shows the optimal pre-computing is BKZ 40 pruning 15 in dimension 104 to130.

HALL OF FAME

Position	Dimension	Euclidean norm	Seed	Contestant	Solution	Algorithm	Subm. Date
1	120	2851	0	Po-Chun Kuo, Michael Schneider	vec	ENUM,BKZ	2011-04-6
2	116	2825	0	Po-Chun Kuo, Michael Schneider	vec	ENUM,BKZ	2011-04-1
3	114	2778	0	Po-Chun Kuo, Michael Schneider	vec	ENUM,BKZ	2011-03-21
4	112	2715	0	Yuanmi Chen and Phong Nguyen	vec	Other	2011-03-30
5	112	2748	0	Po-Chun Kuo	vec	ENUM,BKZ	2011-02-17
6	112	2781	0	Yuanmi Chen and Phong Nguyen	vec	Other	2010-06-5
7	110	2699	0	Yuanmi Chen and Phong Nguyen	vec	Other	2010-05-28
8	108	2508	0	Yuanmi Chen and Phong Nguyen	vec	Other	2010-06-16
9	108	2755	0	Yuanmi Chen and Phong Nguyen	vec	Other	2010-05-30
10	107	2724	8	Po-Chun Kuo, Michael Schneider	vec	ENUM,BKZ	2011-03-12
11	107	2756	4	Michael Schneider, Özgür Dagdelen, Jan Reichelt	vec	ENUM,BKZ	2011-02-14
12	106	2692	0	Po-Chun Kuo	vec	ENUM,BKZ	2011-02-2
13	106	2704	0	Yuanmi Chen and Phong Nguyen	vec	Other	2010-05-26
14	104	2644	0	Urs Wagner	vec	Other	2010-11-
15	104	2668	0	Yuanmi Chen an			

http://www.latticechallenge.org/svp-challenge/

Our Record

- No. 1, dimension 120, seed 0
  - 2300 US dollars paid to Amazon
    - Rent 64 machines about 14.4 hours
  - Machine type
    - Cluster GPU Quadruple Extra Large Instance
    - 2 x Intel Xeon X5570 CPUs
    - 22 GB of memory
    - 2 x NVIDIA Tesla “Fermi” M2050 GPUs
    - 1690 GB of instance storage
    - costs 2.5 US dollars per hour
    - http://aws.amazon.com/ec2/instance-types/

Concluding Remarks

- **Empirical validation of GNR'10 extreme pruning**
- **Parallelize and implement GNR'10 extreme pruning** on GPU and in Cloud
- **Extend the pruning idea to pre-computing and flexible bounding function**
- **New security level measure**
- **Estimate the cost of solving ASVP instances of SVP Challenge in higher dimensions**

- Thank you

- Q & A ?