Fast Force-Directed/Simulated Evolution Hybrid for Multiobjective VLSI Cell Placement

Dr. Sadiq M. Sait
Dept. of Comp. Engineering,
King Fahd University of Petroleum & Minerals,
Dhahran, Saudi Arabia.
sadiq@ccse.kfupm.edu.sa

Junaid Asim Khan
Dept. of Elect. & Comp. Engineering,
The University of British Columbia,
Vancouver, BC, Canada.
junaidk@ece.ubc.ca

Introduction

- Standard Cell Placement
  - Assigning modules to locations on the silicon surface
  - An Intractable problem
  - Numerous Design Objectives, wire-length, power dissipation, delay etc.
- One Solution
  - To use Iterative Heuristics e.g., Simulated Annealing, Genetic Algorithm, Tabu Search, Simulated Evolution etc.

Simulated Evolution

- An excellent heuristic for placement
- Both, solution quality, and run time, are better than other approaches
- If used with fuzzy logic, can optimize multiple objectives
- Problem: For large VLSI circuits, needs to be accelerated
  - Solution
    - ÅParallelization
    - ÅHybridize with Force Directed Algorithm

Simulated Evolution, Basic Steps

- Comprises three step
  - Selection
  - Evaluation
  - Allocation

...ted Evolution: Algorithm

Algorithm Simulated_Evolution(B, $Sol_{initial}$, Stopping Criteria)

B = Bias Value   Sol = Complete Solution

m_i = Module I g_i = Godness of m_i

ALLOCATE(m_i, Sol_i) = Function to allocate m_i in partial solution Sol_i

Begin

Repeat

S = {}

EVALUATION:

ForEach m_i in Sol evaluate g_i

SELECTION:

ForEach m_i in Sol **DO**

begin

IF Random > min (g_i,1)

THEN

begin

S = S U m_i

Remove m_i from Sol

end

end

Sort the element of S

ALLOCATION:

ForEach m_i in S **DO**

begin

ALLOCATE (m_i, Sol_i)

end

Until Sopping Condition is satisfied

Return Best Solution

End (Simulated Evolution)

Evaluation

Selection

Allocation

- Previous Approaches (e.g. BLFSE)
  1. Sort selected cells in descending order of their goodness
  2. Pick the top of the list cell
  3. Swap its location with other cells in the list
  4. Accept the best swap
  5. Remove the cell from sorted list
  6. Go to 2
- Problem
  - $O(n^2)$ time complexity for Allocation
  - n is the number of selected cells
  - In VLSI n is too large and hence a $O(n^2)$ is not practical

Force-directed Allocation

- Solution
  - Use Force-directed heuristic to find best x and y locations for a cell
  - y position indicates the best row, the row nearest to y position is selected, which satisfies the width constraint
  - x position indicates the exact location of the cell in the selected row
  - The selected x position may replace some already well placed cell and hence introduce hill climbing
- Benefit
  - Needs only O(n) time
  - Because x and y locations can be found in O(1)

Force-directed ALLOCATION

- Method

$$F_i = \sum_j w_{ij} \cdot d_{ij} = 0$$

$$\Rightarrow \sum_j w_{ij} \cdot (x_j - x_i) = 0 \;\; \& \sum_j w_{ij} \cdot (y_j - y_i) = 0$$

$$\Rightarrow x_i = \frac{\sum_j w_{ij} \cdot x_j}{\sum_j w_{ij}} \qquad \& \;\; y_i = \frac{\sum_j w_{ij} \cdot y_j}{\sum_j w_{ij}}$$

- Problem
  - To find w_{ij} that satisfies multiple objectives
- Solution
  - Use Fuzzy Logic

Fuzzy weights

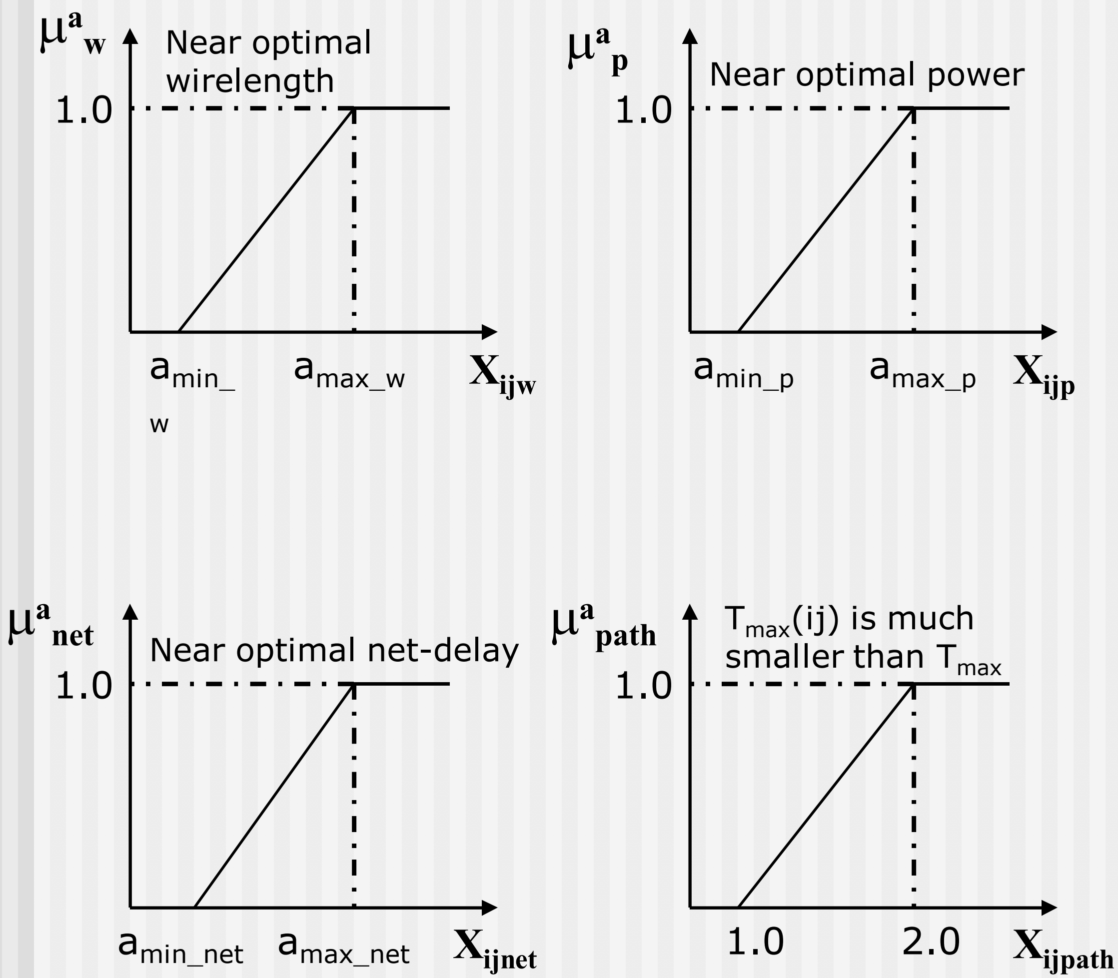
- Following fuzzy rule is used to find weights
  - **IF** a net is good in wire-length AND good in power AND good in delay

  THEN it has a low weight
- A goodness g_{ij} for the net ij in the range [0,1] is found and then w_{ij} is calculated as

$$w_{ij} = 1 \text{ ï } g_{ij}$$

- Higher the goodness lower is the weight

Fuzzy weights (Membership functions)



$$X_{ijw} = \frac{l_{ij}^{*}}{l_{ij}}$$

$$X_{ijp} = \frac{l_{ij}^{*}}{(1+S_{ij})\cdot l_{ij}}$$

$$X_{ijnet} = \frac{ID_{ij}^{*}}{ID_{ij}}$$

$$X_{ijp} = \frac{T_{\max}}{T_{\max}(ij)}$$

Fuzzy weights

- Using AND and OR like Fuzzy aggregation functions and the fuzzy rule we calculate g_{ij} as follows

$$g_{ij} = \mu_{ij}^{a} = 1 - \frac{\sum_{k=w,p,d} \overline{\mu}_{ijk}^{a^2}}{\sum_{k=w,p,d} \overline{\mu}_{ijk}^{a}}$$

where

$$\mu_{ijd}^{a} = \frac{\mu_{ijnet}^{a^2} + \mu_{ijpath}^{a^2}}{\mu_{ijnet}^{a} + \mu_{ijpath}^{a}}$$

- And hence

$$w_{ij} = 1 - g_{ij}$$

Experiments and Results

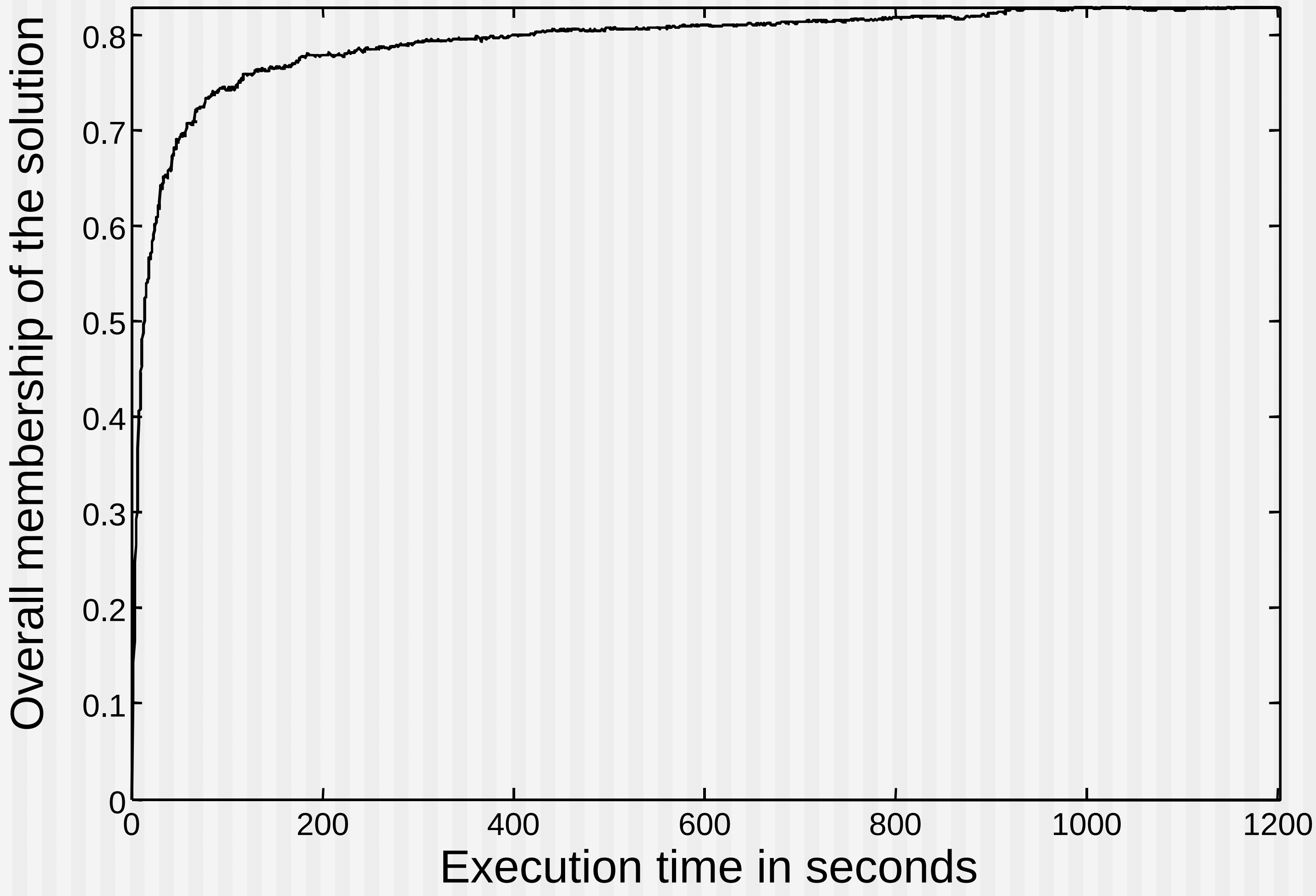
- Fast Fuzzy Force Directed Simulated Evolution (FFSE) is compared with Biasless Fuzzy Simulated Evolution (BLFSE)
- 12 ISCAS benchmark circuits are used
- FFSE is same as BFSE except Allocation
- For BLFSE execution is aborted when there is no improvement in last 500 iterations
- FFSE is run for fixed 5000 iterations
- 0.25 micron technology is used

Experiments and Results

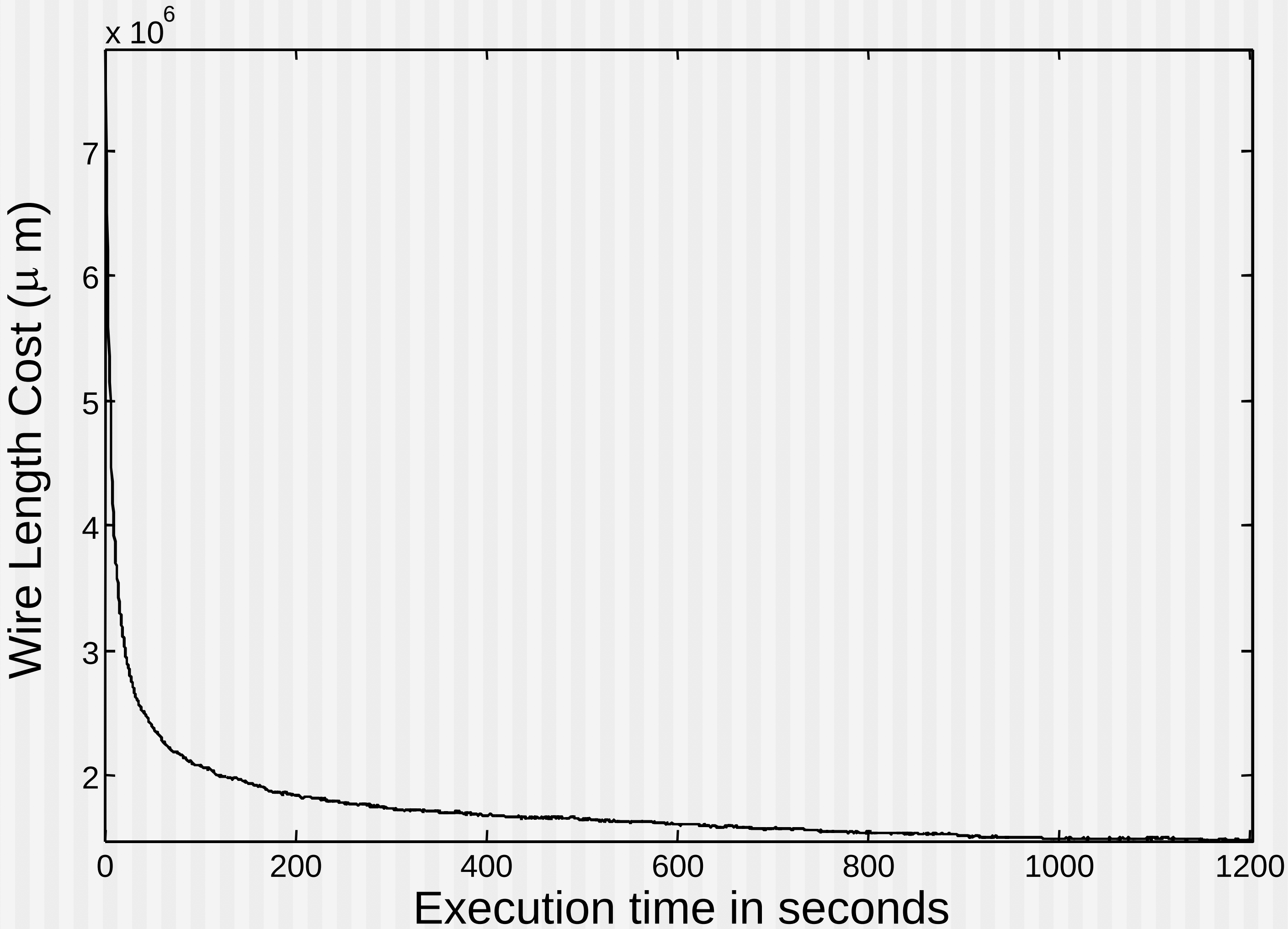
Circuit		BLFSE				FFSE			
	# of cells	L (μm)	P (μm)	D (ps)	T (s)	L (μm)	P (μm)	D (ps)	T (s)
S298	136	4548	915	139	46	4975	999	135	4.8
S386	172	8357	2036	203	117	9422	2169	213	6.8
S832	310	23140	5251	416	192	26112	5863	400	11
S641	433	12811	3072	687	175	12485	2897	674	24
S953	440	29576	5025	223	351	29988	4683	244	17
S1238	540	41318	12303	363	699	41362	12934	377	20
S1196	561	35810	11276	360	613	38282	12363	350	22
S3330	1961	183288	24797	459	5351	163756	24112	483	87
S5378	2993	326840	48360	435	11823	243721	41560	376	149
S9234	5844	UH	UH	UH	UH	655370	114231	908	440
S13207	8651	UH	UH	UH	UH	1339837	144189	1604	885
S15850	10383	UH	UH	UH	UH	1477662	115049	2006	1202

UH: Unreasonably high run time

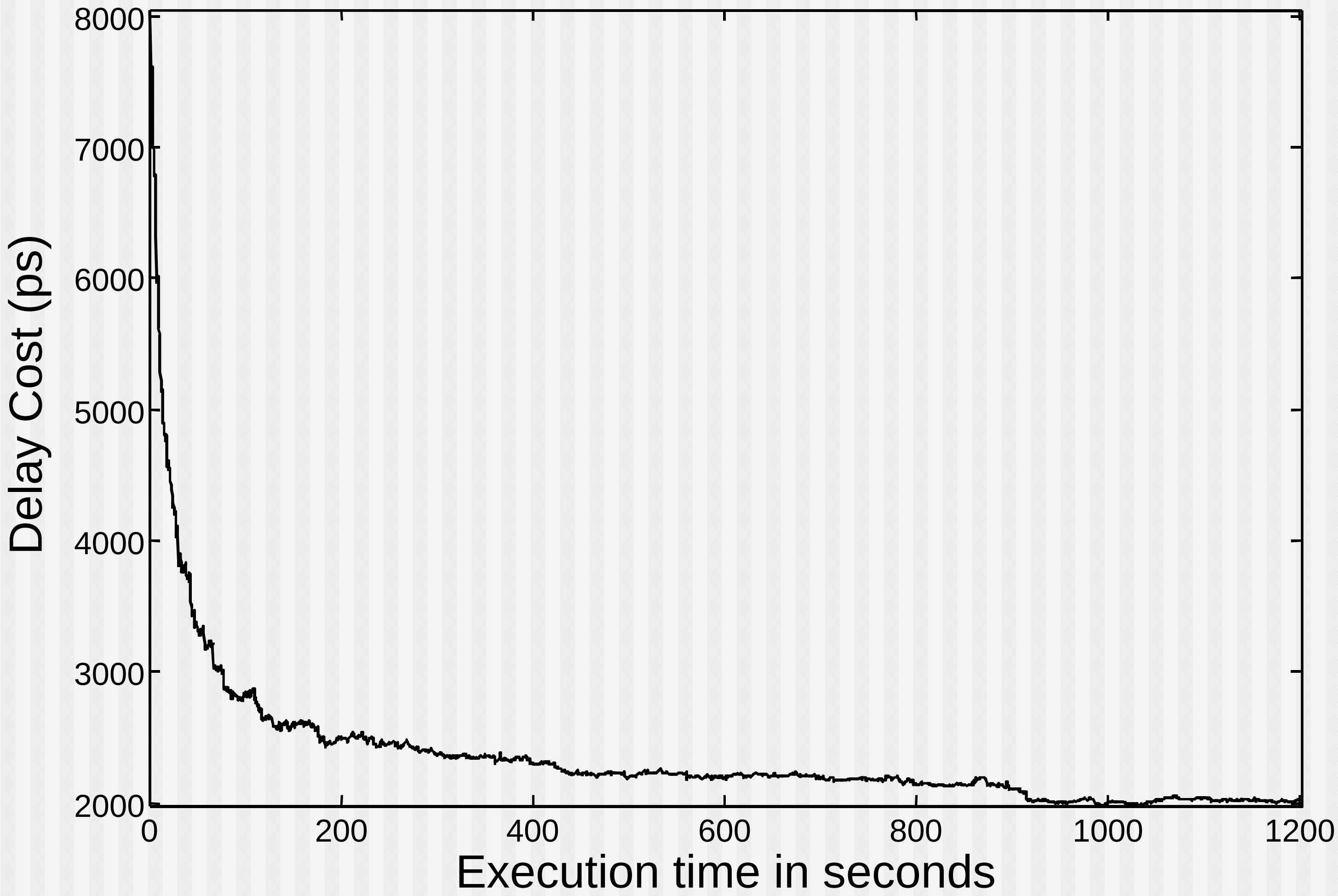
Experiments and Results



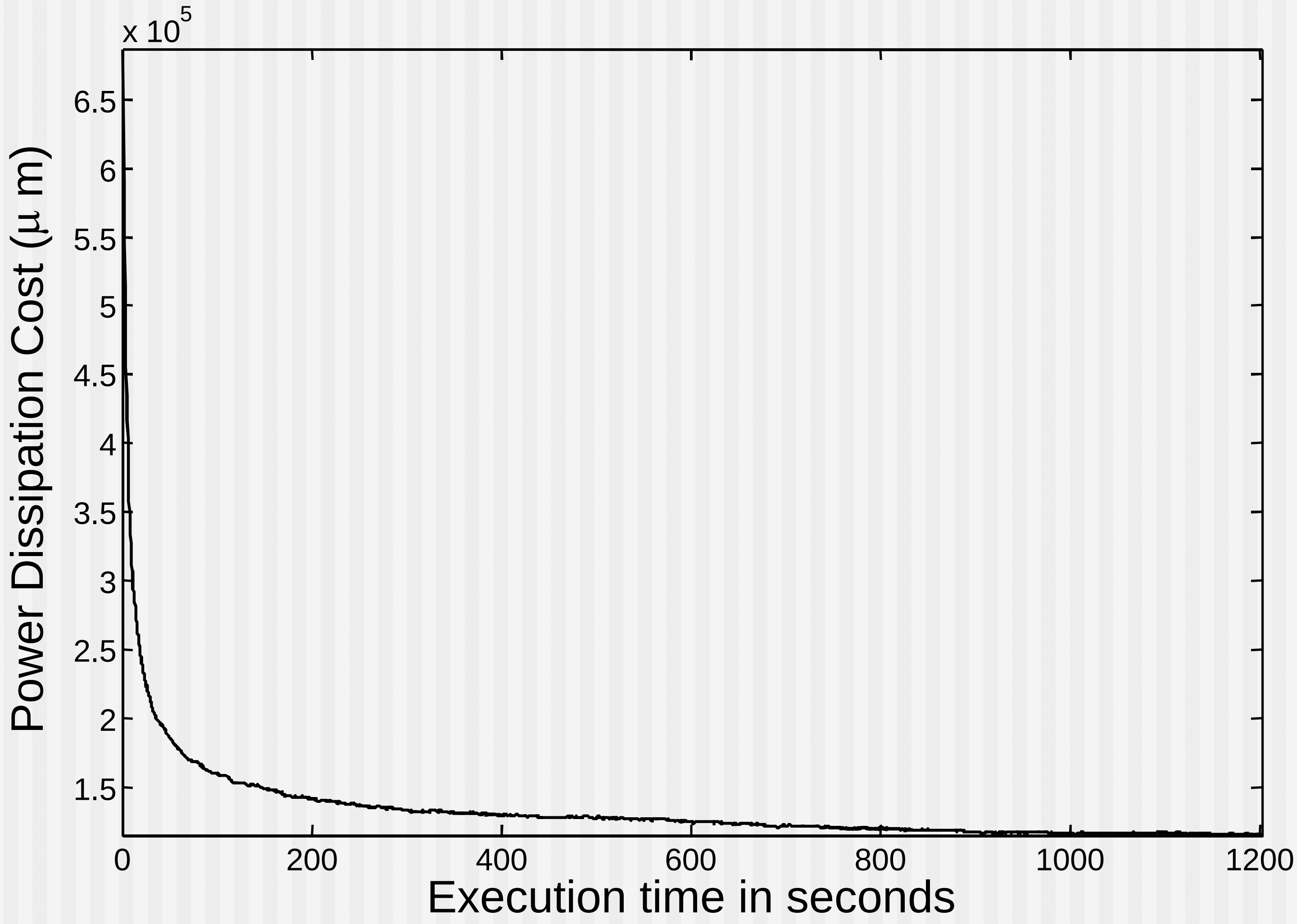
Experiments and Results



Experiments and Results



Experiments and Results



Conclusion

- A Fast Fuzzy Force Directed Simulated Evolution Algorithm for VLSI standard cell placement was proposed
- FFSE is capable of optimizing multiple objectives
- Allocation was speedup from $O(n^2)$ to $O(n)$
- FFSE performs much better than BLFSE in terms of execution time
- There is not much performance degradation in terms of solution quality
- In contrast to BLFSE, FFSE can be used for large circuits

Thank You

Questions

?