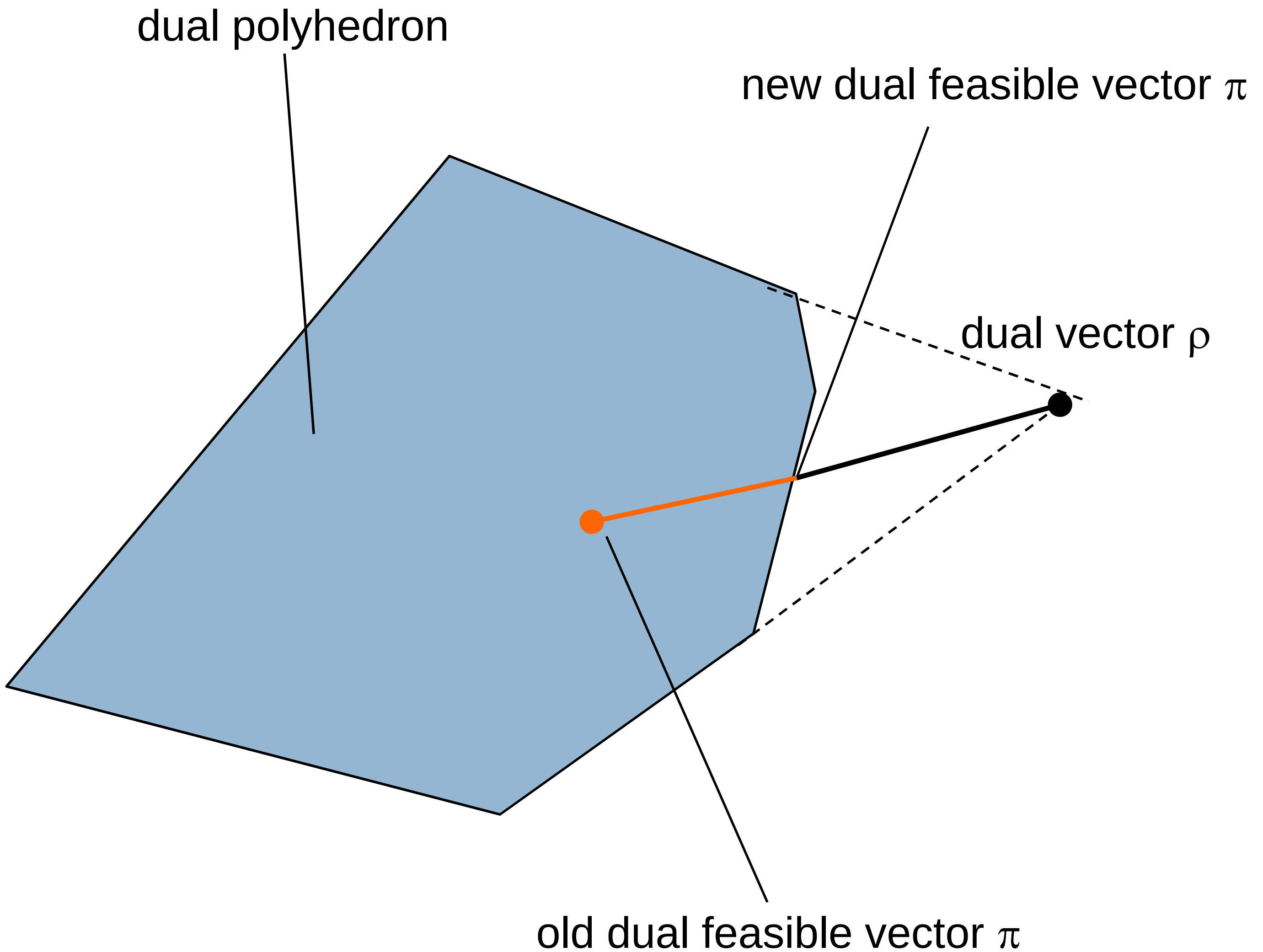
PARALLEL PRIMAL-DUAL WITH COLUMN GENERATION

Diego Klabjan, Northwestern University

Primal-Dual Algorithm

- Started with Dantzig, Ford, and Fulkerson in 1956.
- Primal-dual algorithms
  - Primal step: Solve a primal subproblem.
  - Dual step: Improve the dual feasible solution. Iterate.
- Successfully used in many combinatorial problems
  - Matching algorithms (Edmonds)
  - Minimum cost network flows

Primal-Dual Algorithm



Primal-Dual Algorithm

- The primal-dual algorithm
  - Let ρ be a dual vector and let π be a dual feasible vector.
  - Find a scalar α such that $\alpha\rho+(1-\alpha)\pi$ is a dual feasible vector and the gain in the objective value is maximum.
  - $\pi:=\alpha\rho+(1-\alpha)\pi$.
  - Form a new LP by pricing out columns with best reduced cost based on the new π.
  - Solve the LP and let ρ be an optimal dual solution.
  - Iterate until optimal.
- Developed by H. Jing and Ellis L. Johnson.

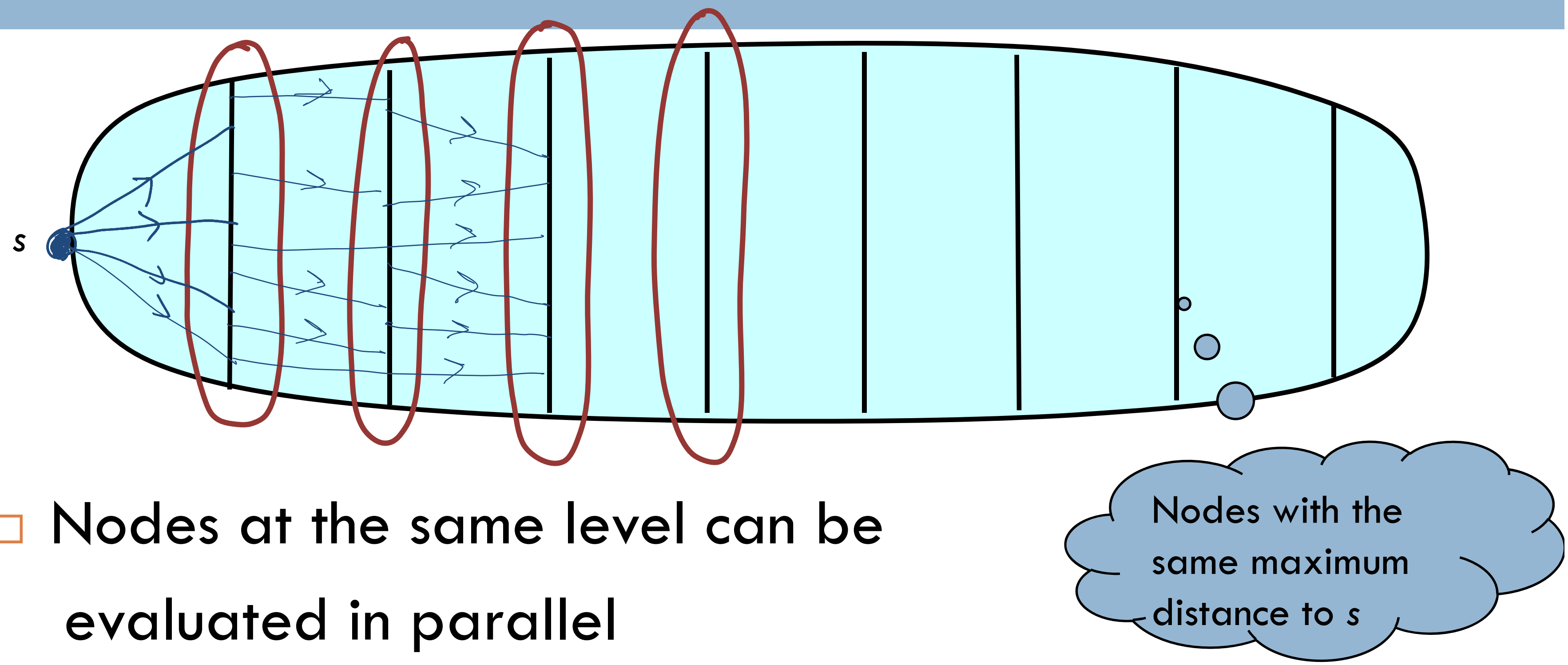
Parallelization

- Solving problems with huge number of columns in short time
  - More problems become tractable
    - Columns spread across machines due to memory limitations
  - Improvements in execution times
- Possible parallelism
  - Parallel pricing strategies
  - Reduce the number of major iterations
  - Using a parallel LP solver

Parallel Pricing

- Parallel depth-first search
  - Relatively easy to parallelize
- Parallel constrained shortest path
  - Interesting and challenging
  - In transportation
    - Many labels
    - Acyclic networks

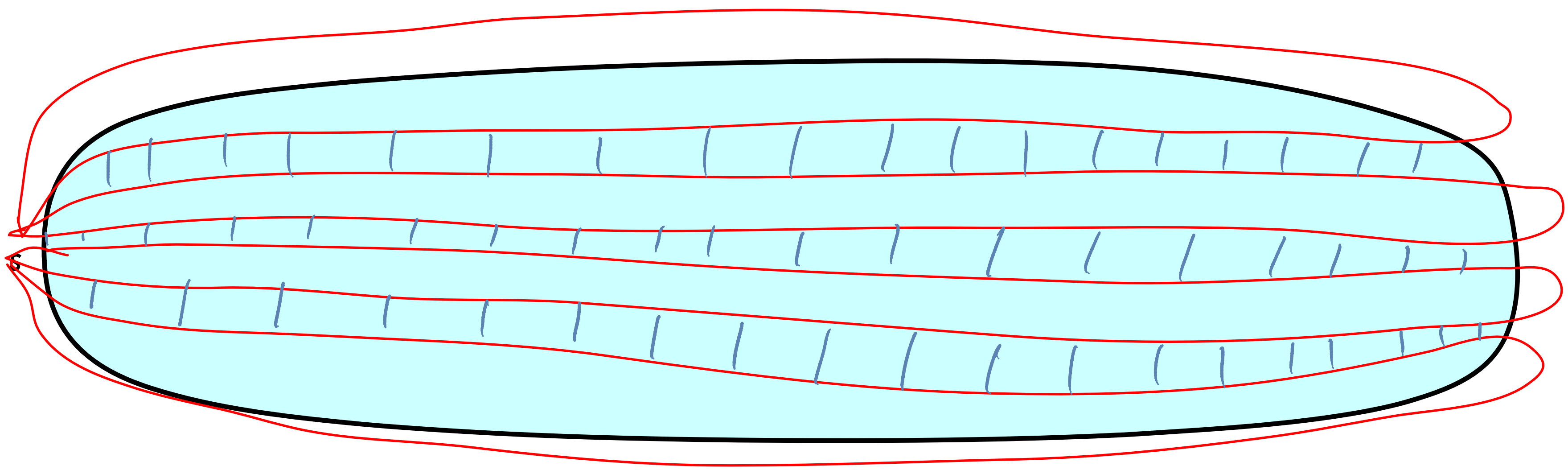
Parallel Layered Algorithm



- Nodes at the same level can be evaluated in parallel
- Embarrassing parallel at each level

Parallel Partitioning

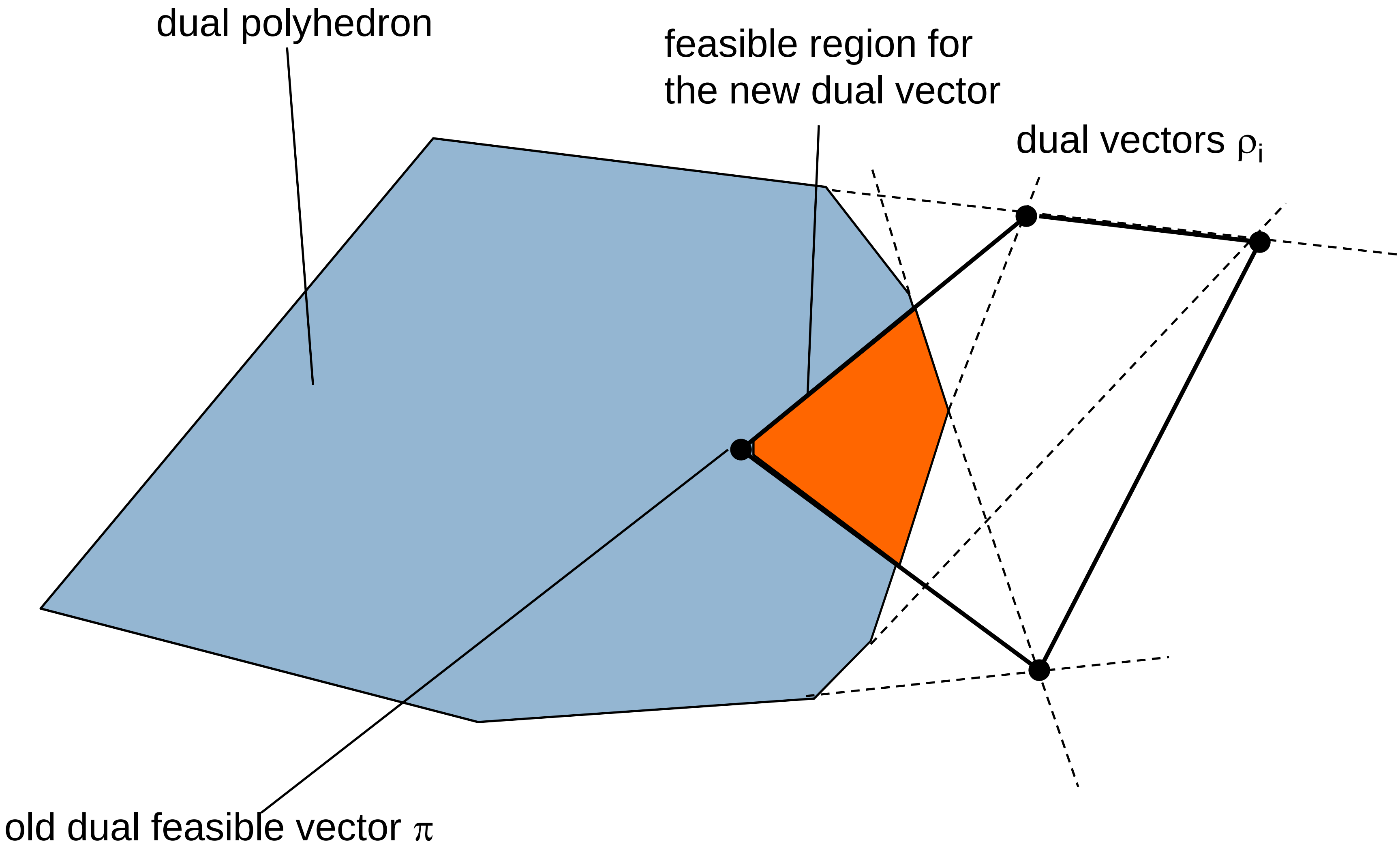
- Partition the network
- Loop
  - Constrained shortest path in each partition
  - Exchange labels at the boundary nodes



Parallel Primal-Dual Algorithm

- The parallel primal-dual algorithm on *p* processors
  - Let $\rho_1, \rho_2, \ldots, \rho_p$ be dual vectors and let π be a dual feasible vector.
  - Find scalars α such that $\alpha_0 \pi + \sum_{i=1}^{p} \alpha_i \rho_i$ is a dual feasible vector and the gain in the objective value is maximum.
  - $\pi := \alpha_0 \pi + \sum_{i=1}^{p} \alpha_i \rho_i$.
  - Form *p* new LPs by controlled randomization based on the new dual feasible vector π.
  - Solve the LPs in parallel. Let ρ_i be an optimal dual solution to the i^{th} LP.
  - Iterate until optimal.

Parallel Primal-Dual Algorithm



Convex Combination of Dual Vectors

- Find scalars α such that $\alpha_0\pi + \sum_{i=1}^{p}\alpha_i\rho_i, \sum_{i=0}^{p}\alpha_i = 1$ is a dual feasible vector and the gain in the objective value is maximum.
- An LP with *p* rows and all columns has to be solved at each iteration to obtain the scalars α.

$$\max \sum_{i=1}^{p} \alpha_i (v_i - v)$$

$$\sum_{i=1}^{p} \alpha_i (rc_{\pi}^{q} - rc_{\rho_i}^{q}) \le rc_{\pi}^{q} \qquad \text{for every column } q$$

$$\sum_{i=1}^{p} \alpha_i \le 1$$

$$a \ge 0$$

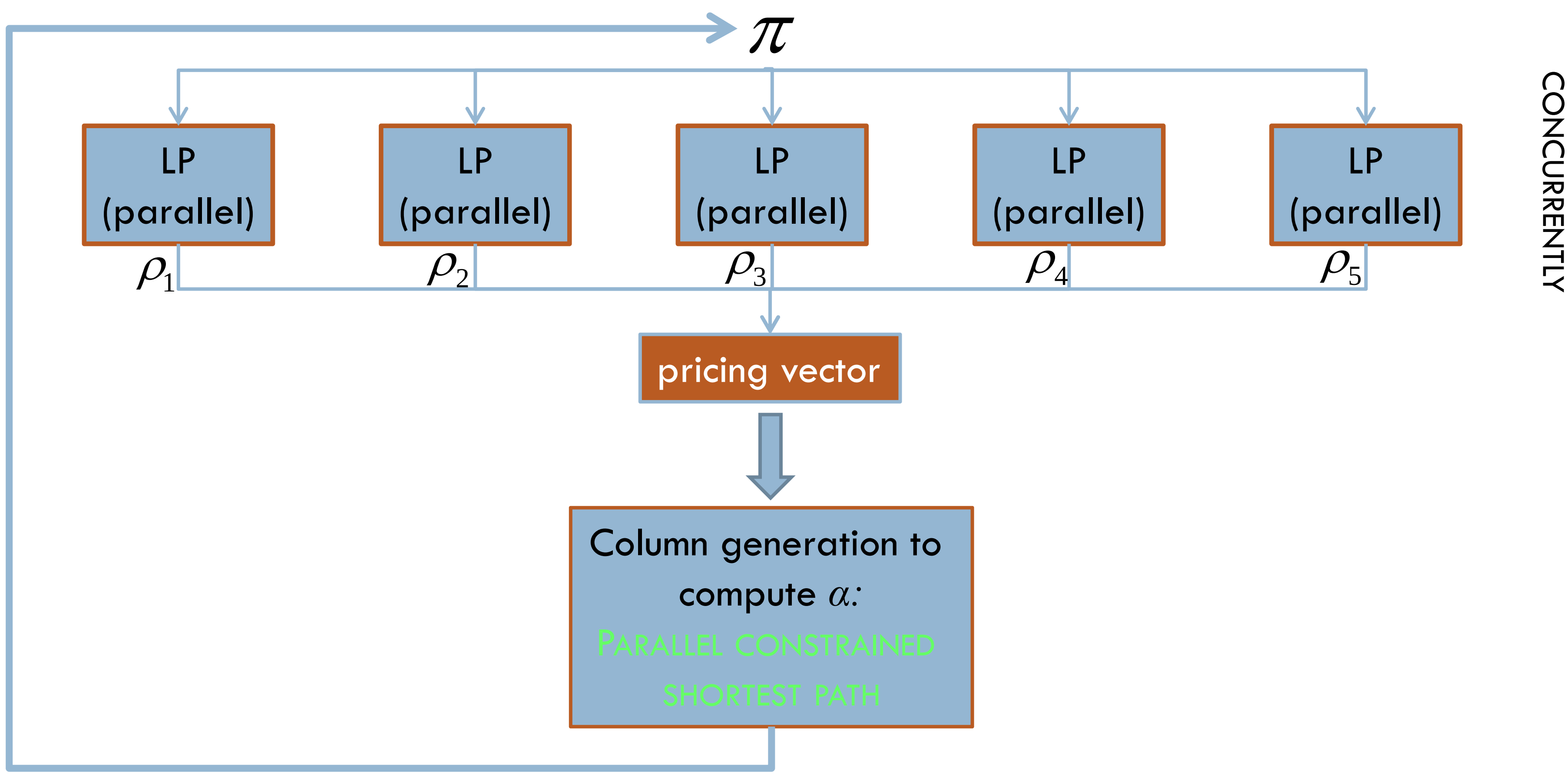
Convex Combination of Dual Vectors

- The reduced cost of a column from the LP
  - Expressed as a reduced cost of the original column with respect to the dual vector

$$(1-\sum_{i=1}^{p}\alpha_i)\pi+\sum_{i=1}^{p}\alpha_i\rho_i$$

  - A parallel constrained shortest path algorithm is employed

Overall Algorithm



Upper Bounds on Variables

- Original LP has upper bounds on variables
- Upper bounds are transferred over to the tiny LP
- Tricky to adjust objective values
  - Doable

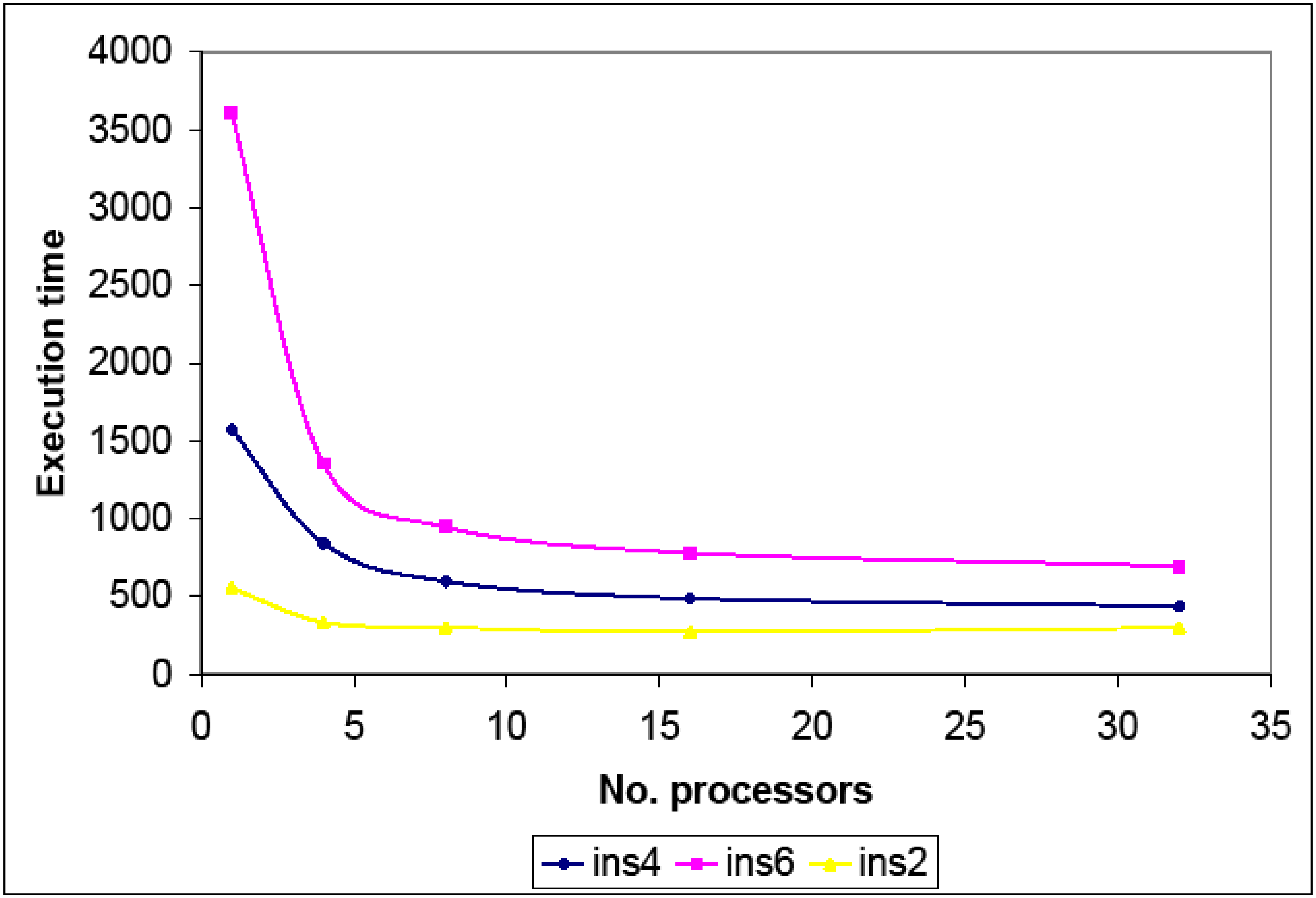
Computational Instances

- Instances
  - Airline crew pairing problems with up to 6 labels
  - Airline aircraft routing problems with up to 3 labels
  - Number of rows ranges from 300 to 3,000
- Networks
  - From 3,000 to 300,000 nodes
  - Number of arcs from 300,000 to 400 million

Computational Architectures

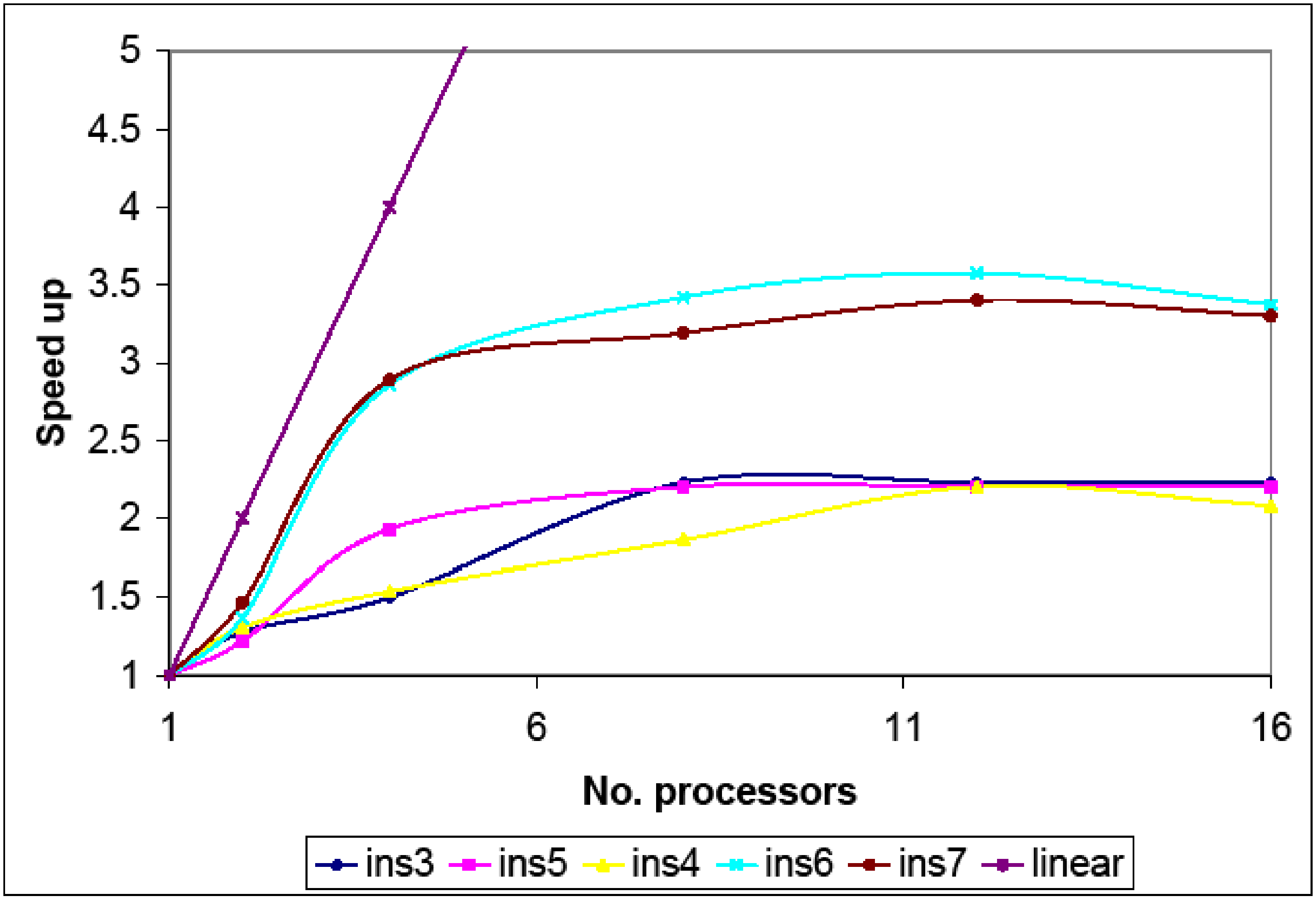
- IHPCL
  - Cluster of 16 personal computers
  - Each one 8-way
  - Dedicated gigabit Ethernet
- NCSA
  - IBM eServers
  - 484 2-way Pentium III
  - 100 Mbit Ethernet

IHPCL



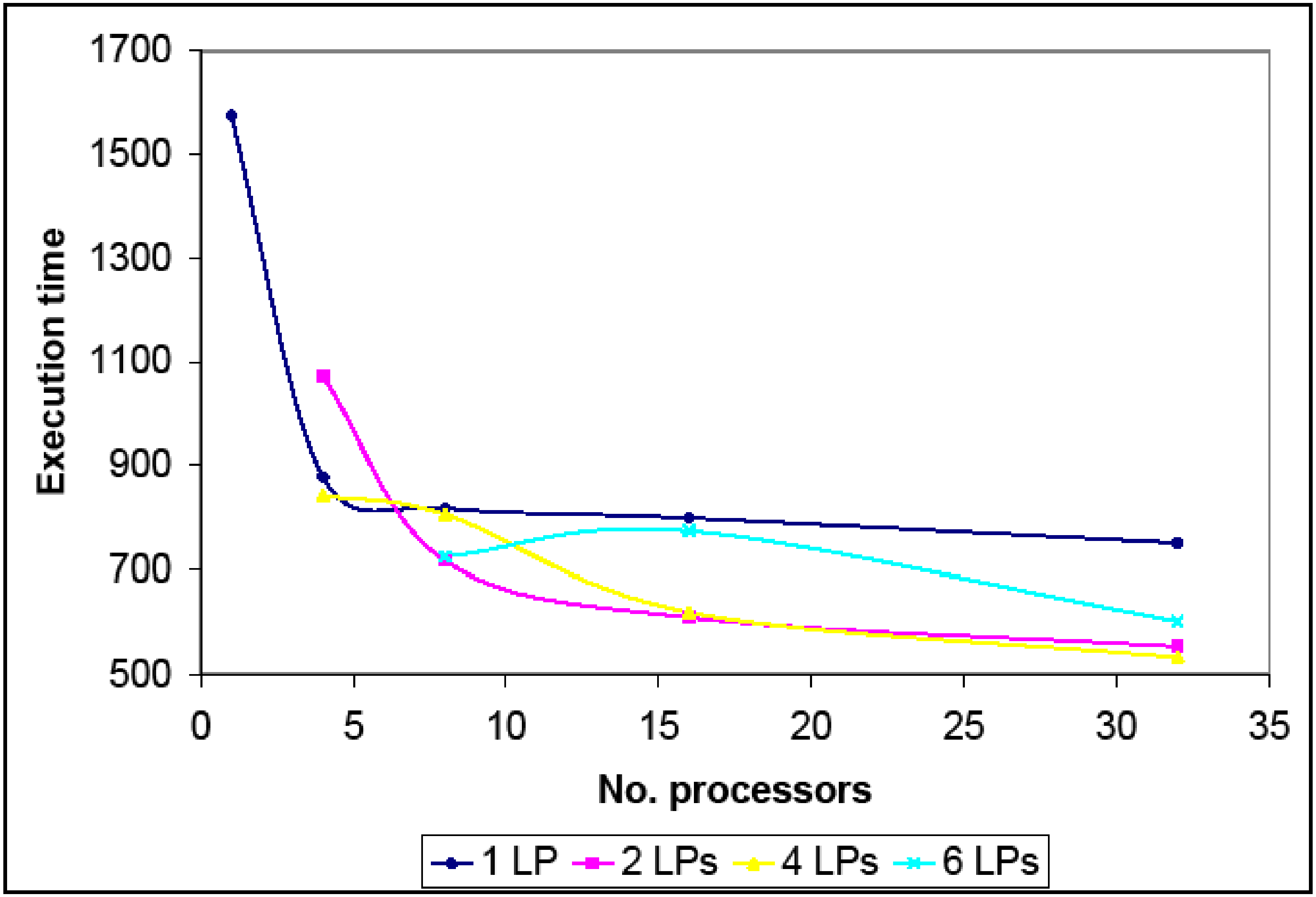
4000
3500
3000
2500
2000
1500
1000
500
0
Execution time
0
5
10
15
20
25
30
35
No. processors
ins4
ins6
ins2

NCSA

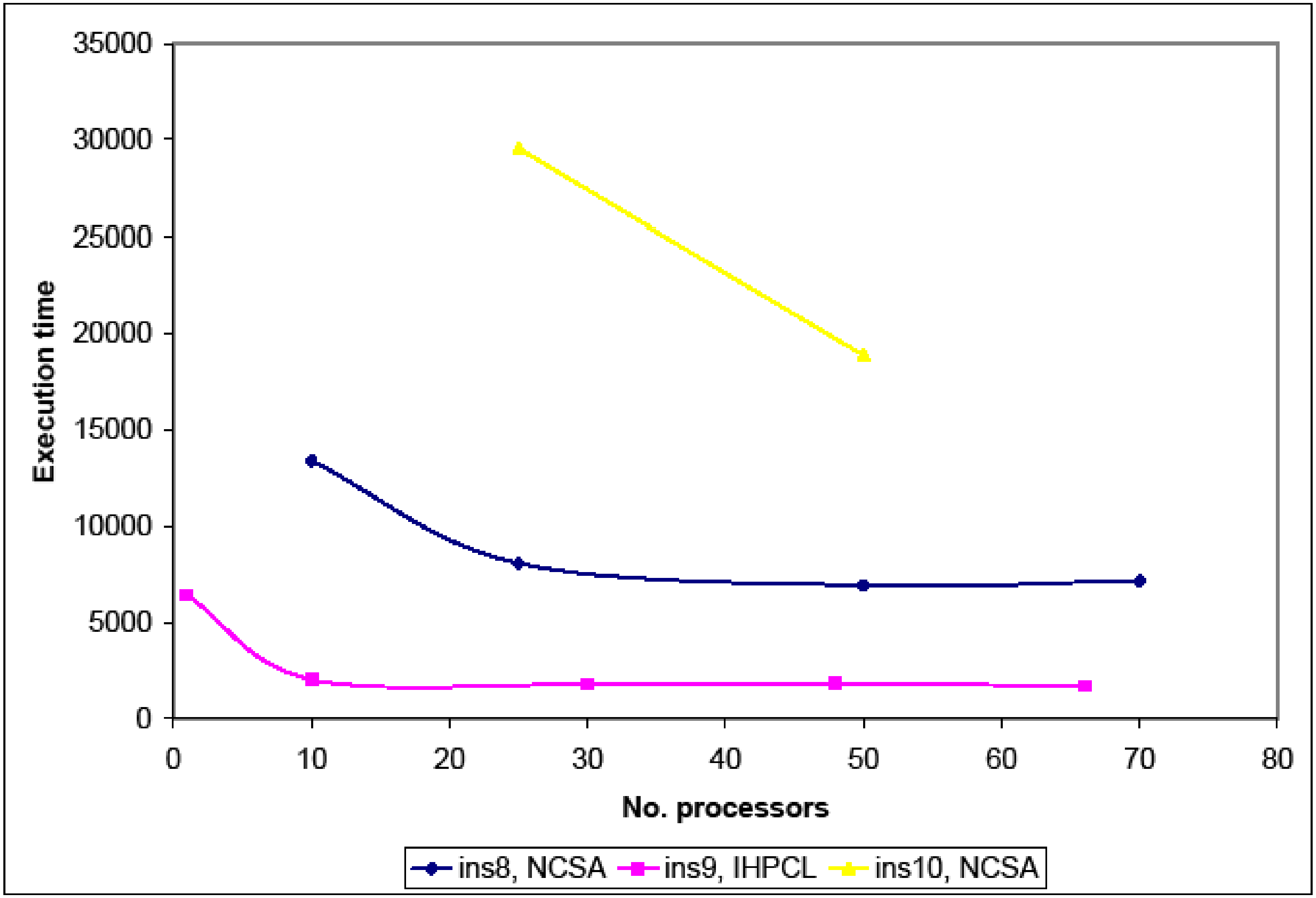


5
4.5
4
3.5
3
2.5
2
1.5
1
Speed up
1
6
11
16
No. processors
ins3
ins5
ins4
ins6
ins7
linear

Number of LPs



Very Large-Scale Instances



Common Theme

- Speed-up
  - Almost linear speed-up up to 8 processors
  - Decent speed-up up to 16 processors
- Elapsed time
  - Elapsed time reductions even up to 30 processors
- Large-scale instances
  - Significant elapsed time reductions even on 50 processors