Parallel Branch-and-Bound for Chemical Engineering Applications: Load Balancing and Scheduling Issues

Chao-Yang Gau and Mark A. Stadtherr*

Department of Chemical Engineering
University of Notre Dame
Notre Dame, IN 46556 USA

VECPAR 2000
Porto, Portugal
June 21–23, 2000

*Fax: (219)631-8366; E-mail: markst@nd.edu

Outline

- Motivation: Reliability in Computing
- Methodologies: Interval Analysis, Branch-and-Prune, Branch-and-Bound
- Examples (Serial Implementation)
  - Phase Stability Analysis
  - Parameter Estimation for Vapor-Liquid Equilibrium (VLE) Models
- Parallel Implementation on a Cluster of Workstations
- Some Performance Results

High Performance Computing

In chemical engineering and other areas of engineering and science, high performance computing is providing the capability to:

- Solve problems faster.
- Solve larger problems.
- Solve more complex problems.

$\Rightarrow$ **Solve problems more reliably.**

Motivation

- In process modeling and other applications, chemical engineers frequently need to solve nonlinear equation systems in which the variables are constrained physically within upper and lower bounds; that is, to solve:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}$$

$$\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U$$

- These problems may:
  - Have multiple solutions
  - Have no solution
  - Be difficult to converge to any solution

Motivation (continued)

- There is also frequent interest in globally minimizing a nonlinear function subject to nonlinear equality and/or inequality constraints; that is, to solve (globally):

$$\min_{\mathbf{x}} \phi(\mathbf{x})$$

subject to

$$\mathbf{h}(\mathbf{x}) = \mathbf{0}$$
$$\mathbf{g}(\mathbf{x}) \geq \mathbf{0}$$
$$\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U$$

- These problems may:
  - Have multiple local minima (in some cases, it may be desirable to find them *all*)
  - Have no solution (infeasible NLP)
  - Be difficult to converge to any local minima

Motivation (continued)

- Floating point arithmetic difficulties may occur
- Example: Rump's problem (1988):

$$\begin{aligned} f(x,y) &= 333.75y^6 \\ &\quad +x^2(11x^2y^2 - y^6 - 121y^4 - 2) \\ &\quad +5.5y^8 + x/2y \end{aligned}$$

- Evaluate $f(x,y)$ for $x = 77617$ and $y = 33096$ using a FORTRAN program.
- All inputs are machine numbers (representable exactly in floating point arithmetic), so errors in function evaluation are due to problems with floating point arithmetic.

Rump’s Problem

- Evaluation on an IBM S/370 using a FORTRAN program
- Single precision

$$f = 1.172603\ldots$$

Rump's Problem

- Evaluation on an IBM S/370 using a FORTRAN program
- Single precision

$$f = 1.172603\ldots$$

- Double precision

$$f = 1.1726039400531\ldots$$

Rump’s Problem

- Evaluation on an IBM S/370 using a FORTRAN program
- Single precision

$$f = 1.172603\ldots$$

- Double precision

$$f = 1.17260394005317\ldots$$

- Extended precision

$$f = 1.172603940053178\ldots$$

Rump's Problem

- Evaluation on an IBM S/370 using a FORTRAN program
- Single precision

$$f = 1.172603\ldots$$

- Double precision

$$f = 1.1726039400531\ldots$$

- Extended precision

$$f = 1.172603940053178\ldots$$

- The correct answer is

$$f = -0.827396059946\ldots$$

Rounding Error and the Patriot Missile

- After the Gulf War, it was determined that (despite contrary publicity during the War) ”the Patriot’s intercept rate [of Scud missiles] could be much lower than ten percent, perhaps even zero.”

- Rounding error in the tracking calculations (due to repeated multiplications by 0.1) was found to be the key problem.

High Performance Computing:

Are We Just Getting the Wrong Answer Faster?

Motivation: Reliability in Computing

- Finding multiple solutions in nonlinear equation solving
- Existence and uniqueness of solutions
- Global vs. local optimization
- Feasibility of NLPs
- Floating point arithmetic problems

Methodologies

- For dealing with these issues there exist methods, based on interval analysis, that, given initial bounds on each variable, can:
  - Find (enclose) any and all solutions to a nonlinear equation system to a desired tolerance
  - Determine that there is no solution of a nonlinear equation system
  - Find the global optimum of a nonlinear objective function

- These methods:
  - Provide a mathematical guarantee of reliability
  - Deal automatically with rounding error, and so also provide a computational guarantee of reliability
  - Represent a particular type of branch-and-prune algorithm (or branch-and-bound for optimization)

Background—Interval Analysis

- A real interval $X = [a, b] = \{x \in \Re \mid a \leq x \leq b\}$ is a segment on the real number line

- An interval vector $\mathbf{X} = (X_1, X_2, ..., X_n)^T$ is an n-dimensional rectangle or “box”.

- Basic interval arithmetic for $X = [a, b]$ and $Y = [c, d]$ is X op $Y = \{x$ op $y \mid x \in X,\ y \in Y\}$

$$X + Y = [a + c, b + d]$$
$$X - Y = [a - d, b - c]$$
$$X \times Y = [min(ac, ad, bc, bd), max(ac, ad, bc, bd)]$$
$$X \div Y = [a, b] \times [1/d, 1/c], \quad 0 \notin Y$$

- For $X \div Y$ when $0 \in Y$, an extended interval arithmetic is available.

- Computed endpoints are *rounded out* to guarantee the enclosure.

Interval Analysis (continued)

- Interval elementary functions (e.g. $\exp(X)$, $\log(X)$, etc.) are also available.

- The interval extension $F(\mathbf{X})$ encloses all values of $f(\mathbf{x})$ for $\mathbf{x} \in \mathbf{X}$. That is, $F(\mathbf{X}) \supseteq \{f(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X}\}$.

- Interval extensions can be computed using interval arithmetic (the “natural” interval extension), or with other techniques

- If a variable occurs more than once in an expression, the natural interval extension may not tightly bound the true range

Interval Analysis (continued)

- Example: $f(x) = x/(x-1)$ evaluated for the interval $X = [2,3]$
- The natural interval extension is

$$\begin{aligned} F([2,3]) &= [2,3]/([2,3]-1) \\ &= [2,3]/[1,2] = [1,3] \end{aligned}$$

- Rearranged $f(x) = x/(x-1) = 1 + 1/(x-1)$, the natural interval extension is

$$\begin{aligned} F([2,3]) &= 1 + 1/([2,3]-1) \\ &= 1 + 1/[1,2] \\ &= 1 + [0.5,1] = [1.5,2] \end{aligned}$$

  which is the true range.
- This is the "dependency" problem. In the first case, each occurrence of x was treated as a independent interval in performing interval arithmetic.

Interval Methodology for Problem Solving

- Interval Newton/Generalized Bisection (IN/GB)
  - Given a system of equations to solve and an initial interval (bounds on all variables):
  - IN/GB can find (enclose) with mathematical and computational certainty either all solutions or determine that no solutions exist. (e.g., Kearfott, 1996; Neumaier, 1990)

- A general purpose approach : requires no simplifying assumptions or problem reformulations

- Why *enclose* solutions?: Even for a simple problem like $10x = 1$, the exact solution ($x = 1/10$) is not a machine-representable number. The best one can do is enclose the solution with a very small interval with machine-representable bounds.

Interval Approach (Cont'd)

Problem: Solve $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ for all roots in initial interval $\mathbf{X}^{(0)}$.

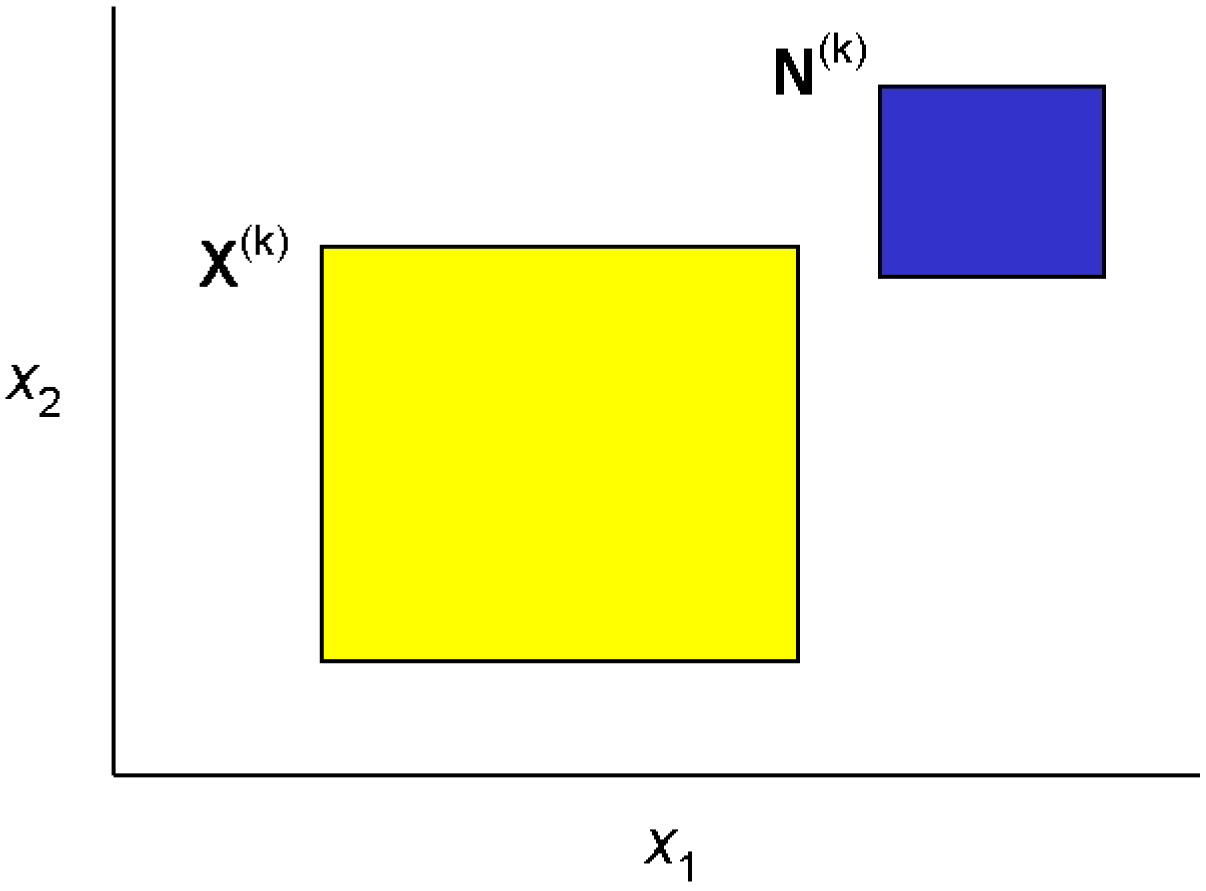
Basic iteration scheme: For a particular subinterval (box), $\mathbf{X}^{(k)}$, arising from some branching (bisection) scheme, perform root inclusion test:

- Compute the interval extension (range) of each function in the system.
- If 0 is not an element of each range, delete (prune) the box.
- If 0 is an element of each range, then compute the *image*, $\mathbf{N}^{(k)}$, of the box by solving the interval Newton equation

$$F'(\mathbf{X}^{(k)})(\mathbf{N}^{(k)} - \mathbf{x}^{(k)}) = -\mathbf{f}(\mathbf{x}^{(k)})$$

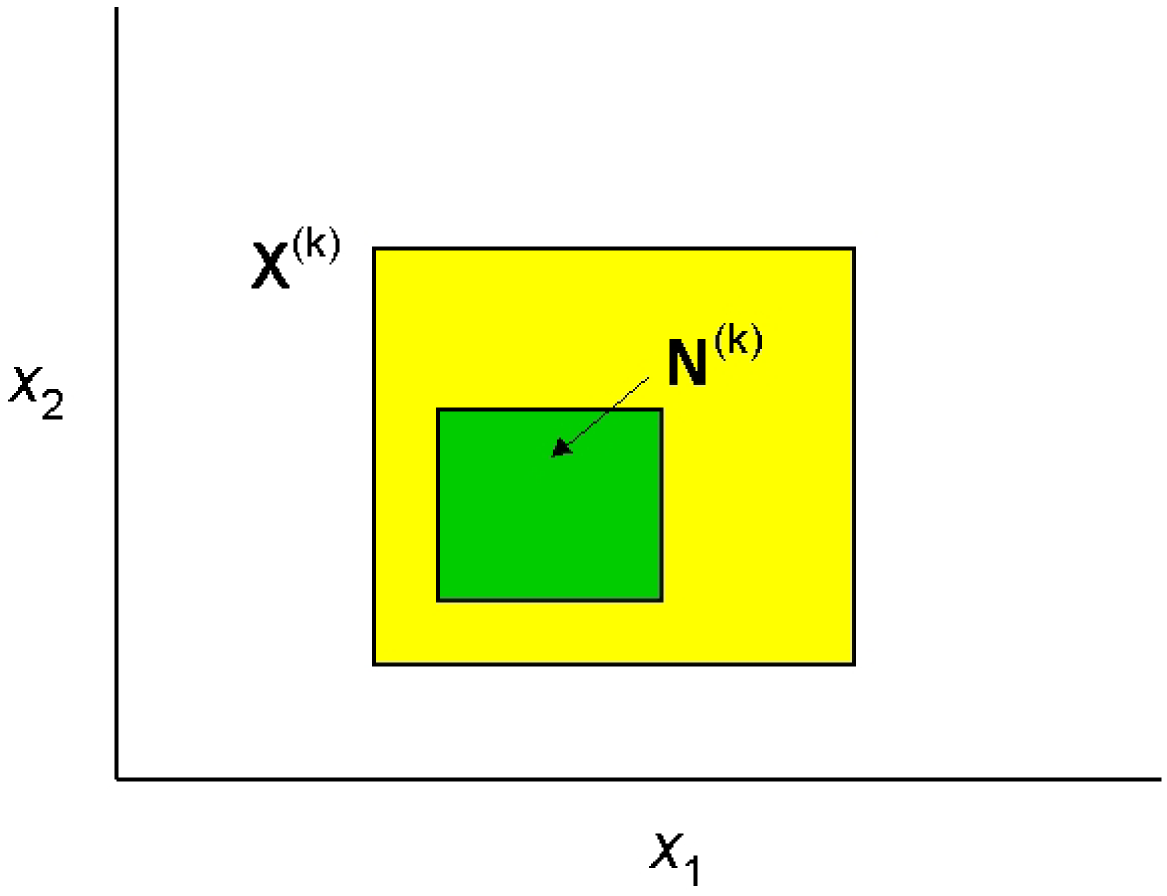
- $\mathbf{x}^{(k)}$ is some point in the interior of $\mathbf{X}^{(k)}$.
- $F'\left(\mathbf{X}^{(k)}\right)$ is an interval extension of the Jacobian of $\mathbf{f}(\mathbf{x})$ over the box $\mathbf{X}^{(k)}$.

Interval Newton Method



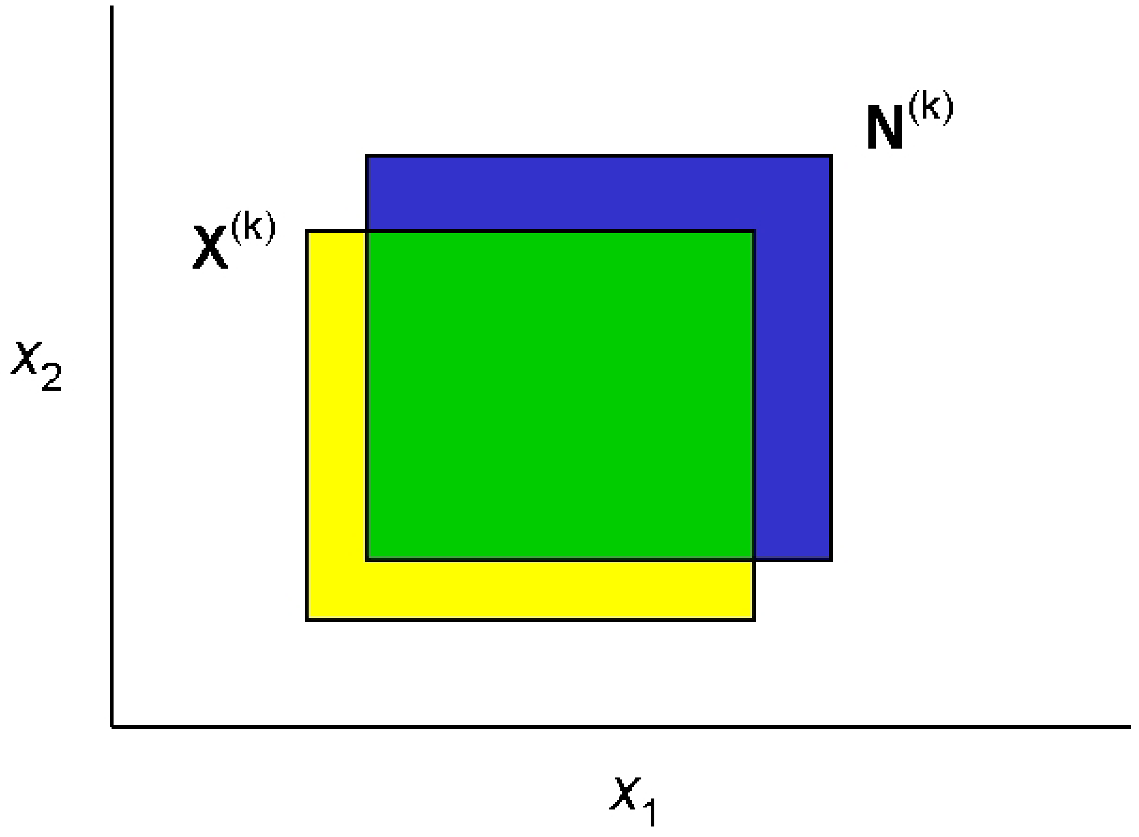
- There is no solution in $\mathbf{X}^{(k)}$.

Interval Newton Method



- There is a *unique* solution in $\mathbf{X}^{(k)}$.
- This solution is in $\mathbf{N}^{(k)}$.
- Point Newton method will converge to solution.

Interval Newton Method



- Any solutions in $\mathbf{X}^{(k)}$ are in intersection of $\mathbf{X}^{(k)}$ and $\mathbf{N}^{(k)}$.
- If intersection is sufficiently small, repeat root inclusion test.
- Otherwise, bisect the intersection and apply root inclusion test to each resulting subinterval.

Interval Approach (Cont'd)

- This is a branch-and-prune scheme on a binary tree.
- No strong assumptions about the function $\mathbf{f}(\mathbf{x})$ need be made.
- The problem $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ must have a finite number of real roots in the given initial interval.
- The method is not suitable if $\mathbf{f}(\mathbf{x})$ is a "black-box" function.
- If there is a solution at a singular point, then existence and uniqueness cannot be confirmed. The eventual result of the IN/GB approach will be a very narrow enclosure that *may* contain one or more solutions.

Interval Approach (Cont'd)

- Can be extended to global optimization problems.
- For unconstrained problems, solve for stationary points
- For constrained problems, solve for KKT points (or more generally for Fritz-John points)
- Add an additional pruning condition:
  - Compute interval extension (range) of objective function.
  - If its lower bound is greater than a known upper bound on the global minimum, prune this subinterval since it cannot contain the global minimum.
- This is a branch-and-bound scheme on a binary tree.

Phase Stability Analysis

- Will a mixture (feed) at a given T, P, and composition $\mathbf{z}$ split into multiple phases?
- A key subproblem in determination of phase equilibrium, and thus in the design and analysis of separation operations.
- Using tangent plane analysis, can be formulated as a minimization problem, or as an equivalent nonlinear equation solving problem.
- Equation system to be solved may have trivial and/or multiple roots (optimization problem has multiple local optima).
- Conventional techniques may fail to converge, or converge to false or trivial solutions.

Tangent Plane Analysis

- A phase at given T, P, and feed composition $\mathbf{z}$ is not stable (and may split) if the Gibbs energy of mixing vs. composition surface

  $$m(\mathbf{x}, v) = \Delta g_{mix} = \Delta \hat{G}_{mix}/RT$$

  ever falls below a plane tangent to the surface at $\mathbf{z}$

  $$m_{tan}(\mathbf{x}) = m(\mathbf{z}, v_{\mathbf{z}}) + \sum_{i=1}^{n} \left(\frac{\partial m}{\partial x_i} \right) \Bigg|_{\mathbf{z}} (x_i - z_i)$$

- That is, if the *tangent plane distance*

  $$D(\mathbf{x}, v) = m(\mathbf{x}, v) - m_{tan}(\mathbf{x})$$

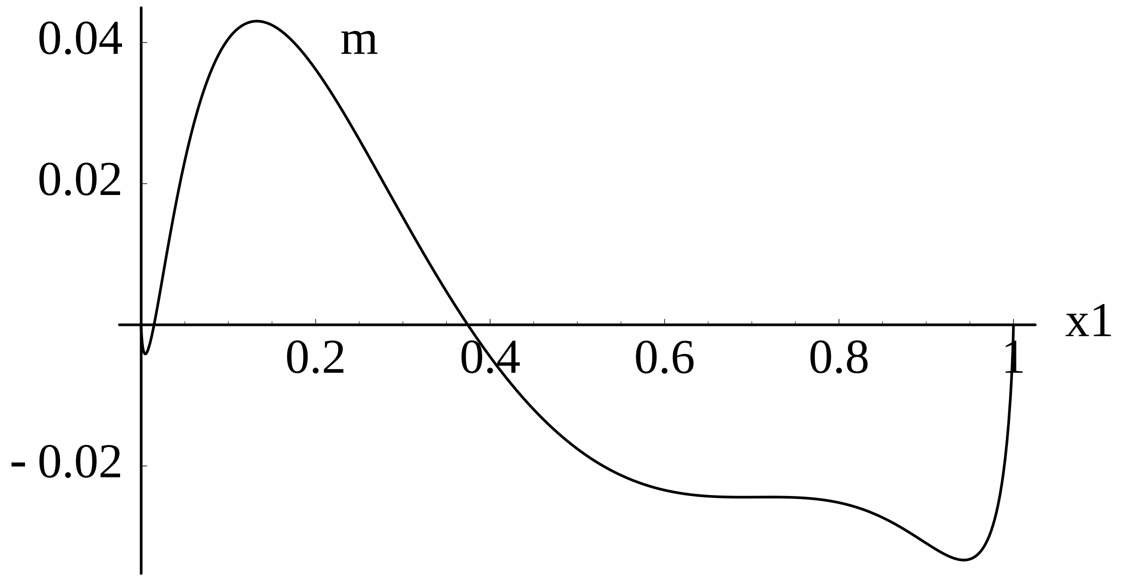
  is negative for any composition $\mathbf{x}$, the phase is not stable.

- In this context, “not stable” refers to both the metastable and classically unstable cases.

Example

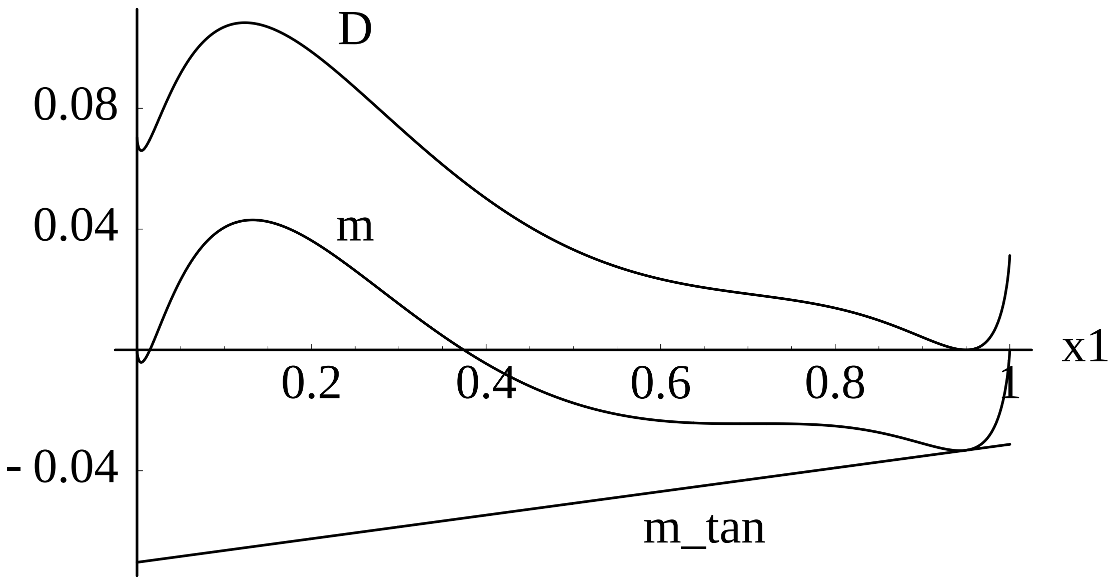
n-Butyl Acetate—Water, NRTL Model

Gibbs energy of mixing m vs. x_1



Example (continued)

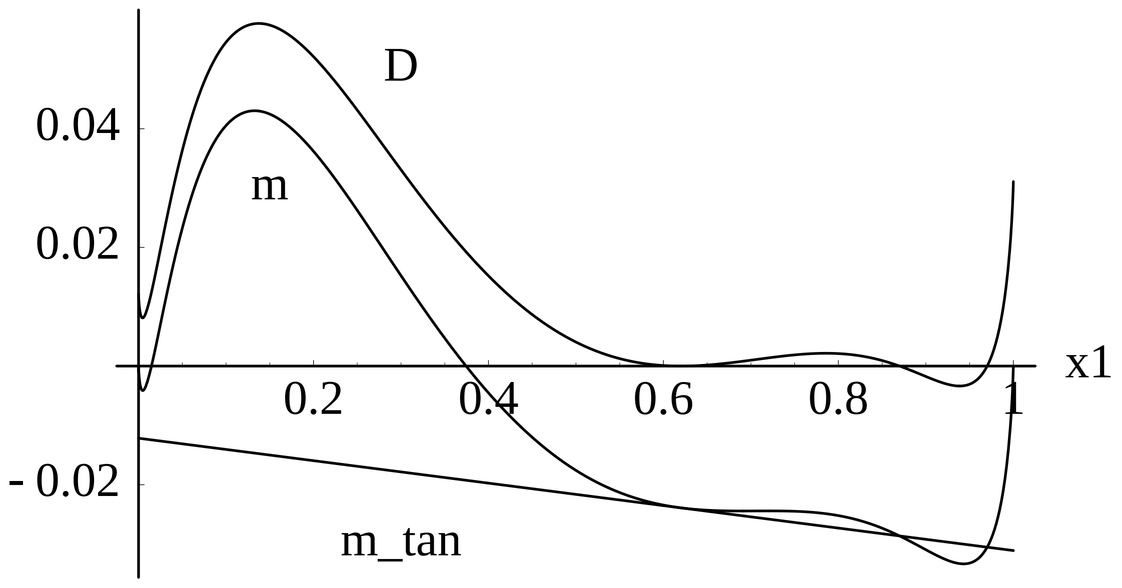
Feed composition z_1 = 0.95



Phase of this composition is stable (D is never negative).

Example (continued)

Feed composition z_1 = 0.62



Phase of this composition is not stable and can split (D becomes negative).

Optimization Formulation

- To determine if D ever becomes negative, determine the minimum of D and examine its sign
$$\min_{\mathbf{x},v} \; D(\mathbf{x}, v)$$
subject to
$$1 - \sum_{i=1}^{n} x_i = 0$$
$$EOS(\mathbf{x}, v) = 0$$

- Trivial local optimum (minimum or maximum) at the feed composition $\mathbf{x} = \mathbf{z}$; may be multiple nontrivial optima. Need technique <u>guaranteed</u> to find the <u>global</u> minimum.

Equation Solving Formulation

- Stationary points of the optimization problem can be found be solving the nonlinear equation system

$$\left[\left(\frac{\partial m}{\partial x_i}\right) - \left(\frac{\partial m}{\partial x_n}\right)\right] - \left[\left(\frac{\partial m}{\partial x_i}\right) - \left(\frac{\partial m}{\partial x_n}\right)\right]_{\mathbf{z}} = 0,$$

$$i = 1, \ldots, n-1$$

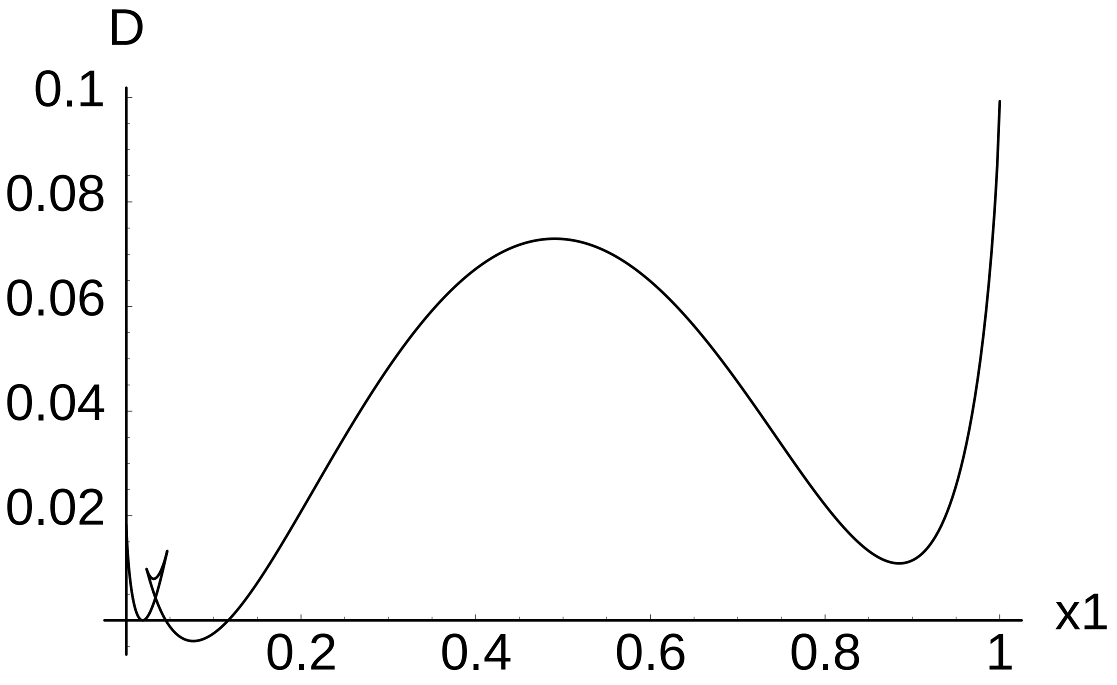
$$1 - \sum_{i=1}^{n} x_i = 0$$

$$EOS(\mathbf{x}, v) = 0$$

- Trivial root at the feed composition $\mathbf{x} = \mathbf{z}$; may be multiple nontrivial roots. Need technique guaranteed to find all the roots.

Example – Phase Stability

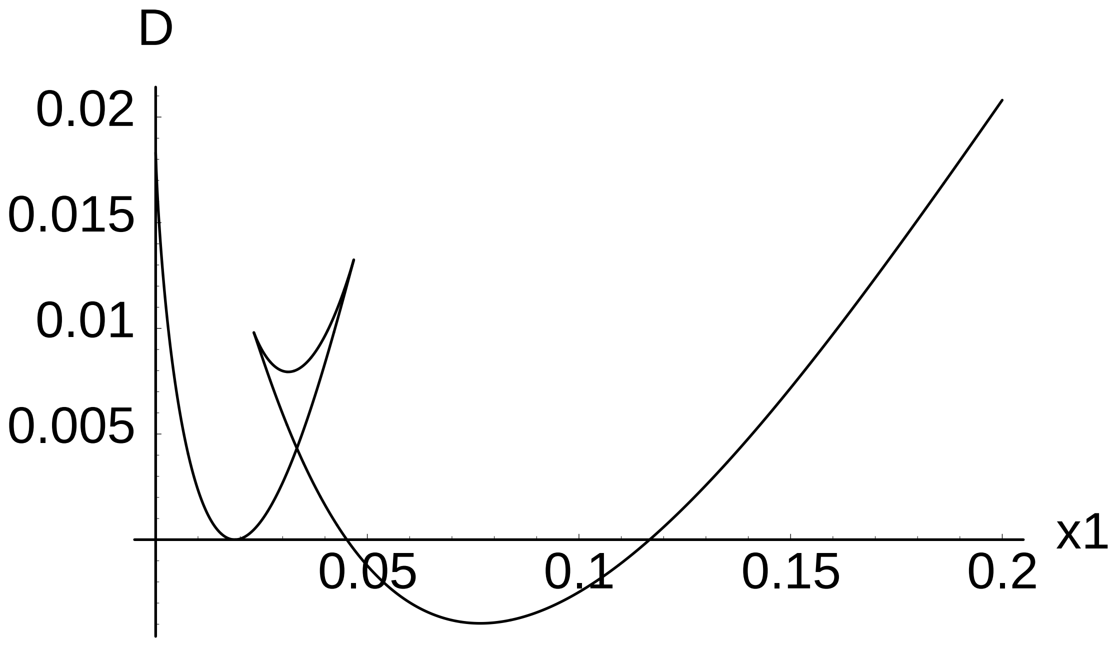
CH_4, H_2S, T = 190 K, P = 40 atm, z_1 = 0.0187, SRK EOS model. Tangent plane distance D vs. x_1



- Five stationary points (four minima, one maximum).
- Standard local methods (e.g. Michelsen, 1982) known to fail (predict stability when system is actually not stable).

Example (continued)

CH_4, H_2S, T = 190 K, P = 40 atm, z_1 = 0.0187, SRK EOS model. Tangent plane distance D vs. x_1 (region near origin)



Example (continued)

- Use interval method to solve the NLE system, finding all the stationary points (Hua et al., 1995)

- Initial interval includes all physically feasible values of mole fraction and molar volume

Feed (z_1, z_2) and CPU time	Stationary Points (roots) $(x_1, x_2, v$ [cm^3/mol])	D
(0.0187, 0.9813)	(0.885, 0.115, 36.6)	0.011
0.20 sec	(0.0187, 0.9813, 207.3)	0.0
	(0.031, 0.969, 115.4)	0.008
	(0.077, 0.923, 64.1)	-0.004
	(0.491, 0.509, 41.5)	0.073

- CPU time on Sun Ultra 2/1300.
- All stationary points easily found, showing the feed to be not stable.
- Presence of multiple real volume roots causes no difficulties.

Parameter Estimation in VLE Modeling

- Goal: Determine parameter values in liquid phase activity coefficient models (e.g. Wilson, van Laar, NRTL, UNIQUAC):

$$\gamma_{\mu i,\mathrm{calc}} = f_i(\mathbf{x}_\mu, \boldsymbol{\theta})$$

- The relative least squares objective is:

$$\phi(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{\mu=1}^{p} \left[\frac{\gamma_{\mu i,\mathrm{calc}}(\boldsymbol{\theta}) - \gamma_{\mu i,\mathrm{exp}}}{\gamma_{\mu i,\mathrm{exp}}} \right]^2 .$$

- Experimental values $\gamma_{\mu i,\mathrm{exp}}$ of the activity coefficients are obtained from VLE measurements at compositions $\mathbf{x}_\mu, \mu = 1, \ldots, p$.

- This problem has been solved for many models, systems, and data sets in the DECHEMA VLE Data Collection (Gmehling *et al.*, 1977-1990).

Parameter Estimation (Cont'd)

- A common approach for solving this problem is to use the gradient of $\phi(\boldsymbol{\theta})$ and to seek the stationary points of $\phi(\boldsymbol{\theta})$ by solving $\mathbf{g}(\boldsymbol{\theta}) \equiv \nabla\phi(\boldsymbol{\theta}) = \mathbf{0}$.

- This system may have many roots, including local minima, local maxima and saddle points.

- To insure that the global minimum of $\phi(\boldsymbol{\theta})$ is found, the capability to find *all* the roots of $\mathbf{g}(\boldsymbol{\theta}) = \mathbf{0}$ is needed. This is provided by the interval technique (IN/GB).

- Interval Newton can be combined with branch-and-bound so that roots of $\mathbf{g}(\boldsymbol{\theta}) = \mathbf{0}$ that cannot be the global minimum need not be found.

Example – Parameter Estimation

- The binary system benzene (1) and hexafluorobenzene (2) was studied.

- Ten problems, each a different data set from the DECHEMA VLE Data Collection were considered.

- The model used was the Wilson equation. This has binary interaction parameters
  $$\Lambda_{12} = (v_2/v_1)\exp(-\theta_1/RT) \text{ and}$$
  $$\Lambda_{21} = (v_1/v_2)\exp(-\theta_2/RT)$$
  where v_1 and v_2 are pure component molar volumes.

- The energy parameters θ_1 and θ_2 must be estimated.

- Parameter estimation results for θ_1 and θ_2 are given in the DECHEMA Collection for all ten problems.

Results

- Each problem was solved using the IN/GB approach to determine the globally optimal values of the θ_1 and θ_2 parameters (Gau et al., 2000).

- These results were compared to those presented in the DECHEMA Collection.

- For each problem, the number of local minima in $\phi(\boldsymbol{\theta})$ was also determined (branch and bound steps were turned off).

- Table 1 compares parameter estimation results for θ_1 and θ_2 with those given in the DECHEMA Collection. New globally optimal parameter values are found in five cases.

Table 1: IN/GB results vs. DECHEMA values

Data Set	Data points	T (^oC)	DECHEMA			IN/GB			No. of Minima	CPU time(s)
			θ_1	θ_2	$\phi(\boldsymbol{\theta})$	θ_1	θ_2	$\phi(\boldsymbol{\theta})$		
1*	10	30	437	-437	0.0382	**-468**	**1314**	**0.0118**	2	15.1
2*	10	40	405	-405	0.0327	**-459**	**1227**	**0.0079**	2	13.7
3*	10	50	374	-374	0.0289	**-449**	**1157**	**0.0058**	2	12.3
4*	11	50	342	-342	0.0428	**-424**	**984**	**0.0089**	2	10.9
5	10	60	-439	1096	0.0047	-439	1094	0.0047	2	9.7
6	9	70	-424	1035	0.0032	-425	1036	0.0032	2	7.9

Data Set	Data points	P (mmHg)	DECHEMA			IN/GB			No. of Minima	CPU time(s)
			θ_1	θ_2	$\phi(\boldsymbol{\theta})$	θ_1	θ_2	$\phi(\boldsymbol{\theta})$		
7*	17	300	344	-347	0.0566	**-432**	**993**	**0.0149**	2	17.4
8	16	500	-405	906	0.0083	-407	912	0.0083	2	14.3
9	17	760	-407	923	0.0057	-399	908	0.0053	1	13.9
10	17	760	-333	702	0.0146	-335	705	0.0146	2	20.5

***:New globally optimal parameters found.**

Discussion

- Does the use of the globally optimal parameters make a significant difference when the Wilson model is used to predict vapor-liquid equilibrium (VLE)?

- A common test of the predictive power of a model for VLE is its ability to predict azeotropes.

- Experimentally this system has two homogeneous azeotropes.

- Table 2 shows comparison of homogeneous azeotrope prediction when the locally optimal DECHEMA parameters are used, and when the global optimal parameters are used.

Table 2: Homogeneous azeotrope prediction

Data Set	$T(^oC) or$ P (mmHg)	DECHEMA			IN/GB		
		x_1	x_2	P or T	x_1	x_2	P or T
1	T=30	0.0660	0.9340	P=107	0.0541	0.9459	P=107
					0.9342	0.0658	121
2	40	0.0315	0.9685	168	0.0761	0.9239	168
					0.9244	0.0756	185
3	50	NONE			0.0988	0.9012	255
					0.9114	0.0886	275
4	50	NONE			0.0588	0.9412	256
					0.9113	0.0887	274
7	P=300	NONE			0.1612	0.8388	T=54.13
					0.9315	0.0685	52.49

- Based on DECHEMA results, one would conclude Wilson is a poor model for this system. But actually Wilson is a reasonable model if the parameter estimation problem is solved correctly.

Other Types of Problems Solved

- Location of azeotropes (Maier *et al.*, 1998, 1999, 2000)
  - Homogeneous
  - Heterogeneous
  - Reactive
- Location of mixture critical points (Stradi *et al.*, 1999)
- Solid-fluid equilibrium (Xu *et al.*, 2000)
- General process modeling problems – up to 163 equations (Schnepper and Stadtherr, 1996)

Parallel Branch-and-Bound Techniques

- Branch-and-Bound (BB) and branch-and-prune (BP) have important applications in engineering and science, especially when a *global* solution is sought
  - analysis of phase behavior
  - process synthesis
  - molecular modeling
  - etc.
- BB and BP involve successive subdivision of the problem domain to create subproblems, thus requiring a tree search process
  - Applications are often computationally intense
  - Subproblems (tree nodes) are independent
  - A natural opportunity for use of parallel computing
- There are various BB and BP schemes; we use an interval Newton/generalized bisection (IN/GB) method.

Parallel BB (cont'd)

- For practical problems, the binary tree that needs to be searched may be quite large.
- The binary trees may be highly irregular, and can result in highly uneven distribution of work among processors and thus poor overall performance (e.g., idle processors).
- Need an effective work scheduling and load balancing scheme to do parallel tree search efficiently.
- Manager-worker schemes (centralized global stack management) are popular but scale poorly due to communication expense and bottlenecks.
- Many implementations of parallel BB have been studied (Kumar et al., 1994; Gendron and Crainic, 1994) for various target architectures.

Work Scheduling and Load Balancing

- Objective: Schedule the workload among processors to minimize communication delays and execution time, and maximize computing resource utilization.

- Use Dynamic Scheduling
  - Redistribute workload concurrently at runtime.
  - Transfer workload from a heavily loaded processor to a lightly loaded one (load balancing).

- Target architecture: Distributed computing on a networked cluster using message passing.
  - Often relatively inexpensive.
  - Uses widely available hardware.

- Use distributed (multiple pool) load balancing.

Distributed Load Balancing

- Each processor locally makes the workload placement decision to maintain the local interval stack and prevent itself from becoming idle.
- Alleviates bottleneck effects from centralized load balancing policy (manager/worker).
- Reduction of communication overhead could provide high scalability for the parallel computation.
- Components of typical schemes
  - Workload state measurement
  - State information exchange
  - Transfer initiation
  - Workload placement
  - Global termination

Components

- Workload state measurement
  - Evaluate local workload using some “work index.”
  - Use stack length: number of intervals (boxes) remaining to be processed.
- State information exchange
  - Communicate local workload state to other “cooperating” processors
  - Selection of cooperating processors defines a virtual network
  - Virtual network: Global (all-to-all), 1-D torus, 2-D torus, etc.
- Transfer initiation
  - Sender initiate
  - Receiver initiate
  - Symmetric (sender or receiver initiate)

Components (cont'd)

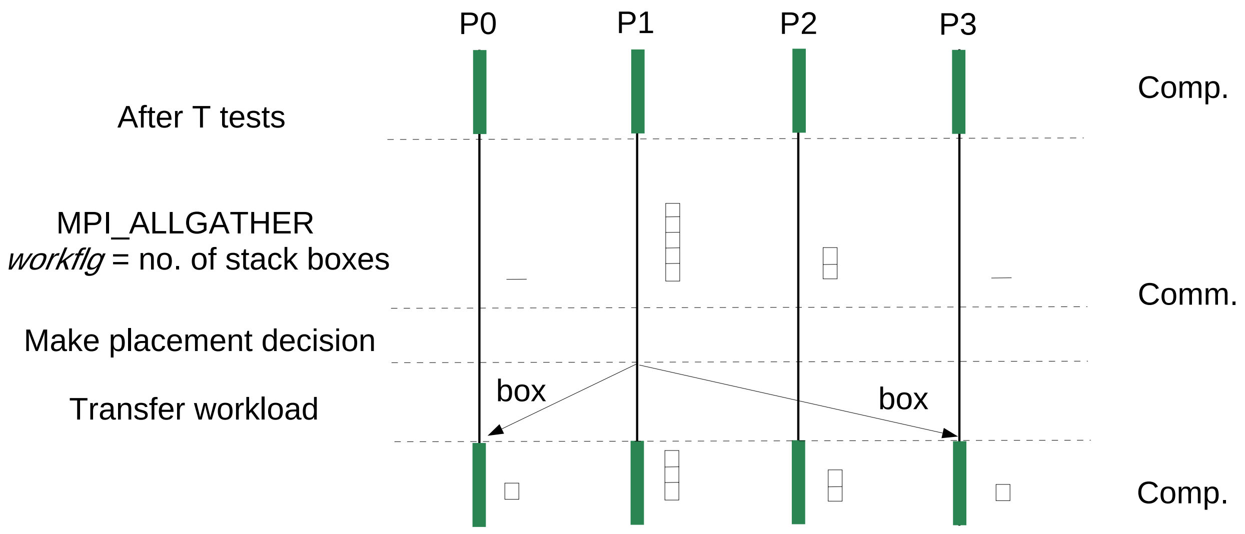
- Workload placement
  - Work-adjusting rule: How to distribute work (boxes) among cooperating processors and how much to transfer
    - Work stealing (e.g., Blumofe and Leiserson, 1994)
    - Diffusive propagation (e.g., Heirich and Taylor, 1995)
    - Etc.
  - Work-selection rule: Which boxes should be transferred
    - Breadth first
    - Best first (based on the lower bound value)
    - Depth first
    - Various heuristics
- Global termination
  - Easy to detect with synchronous, all-to-all communication
  - For local and/or asynchronous communication, use Dijkstra's token algorithm.

Parallel Implementations

- Three types of strategies were implemented.
  - Synchronous Work Stealing (SWS)
  - Synchronous Diffusive Load Balancing (SDLB)
  - Asynchronous Diffusive Load Balancing (ADLB)
- These are listed in order of likely effectiveness.
- All were implemented in Fortran-77 using LAM (Local Area Multicomputer) MPI (Laboratory for Scientific Computing, University of Notre Dame).

Synchronous Work Stealing

- Periodically exchange workload information ($workflg$) and any improved upper bound value (for optimization) using synchronous global (all-to-all) blocking communication.
- Once idle, steal one interval (box) from the processor with the heaviest work load (receiver initiate)
- Difficulties
  - Large network overhead (global, all-to-all)
  - Idle time from process synchronism and blocking communication



Synchronous Diffusive Load Balancing

- Use *local* communication: Processors periodically exchange work state and units of work with their immediate neighbors to maintain their workload.

- Typical workload adjusting scheme (symmetric initiation):

$$u(j) = 0.5[workflg(i) - workflg(j)]$$

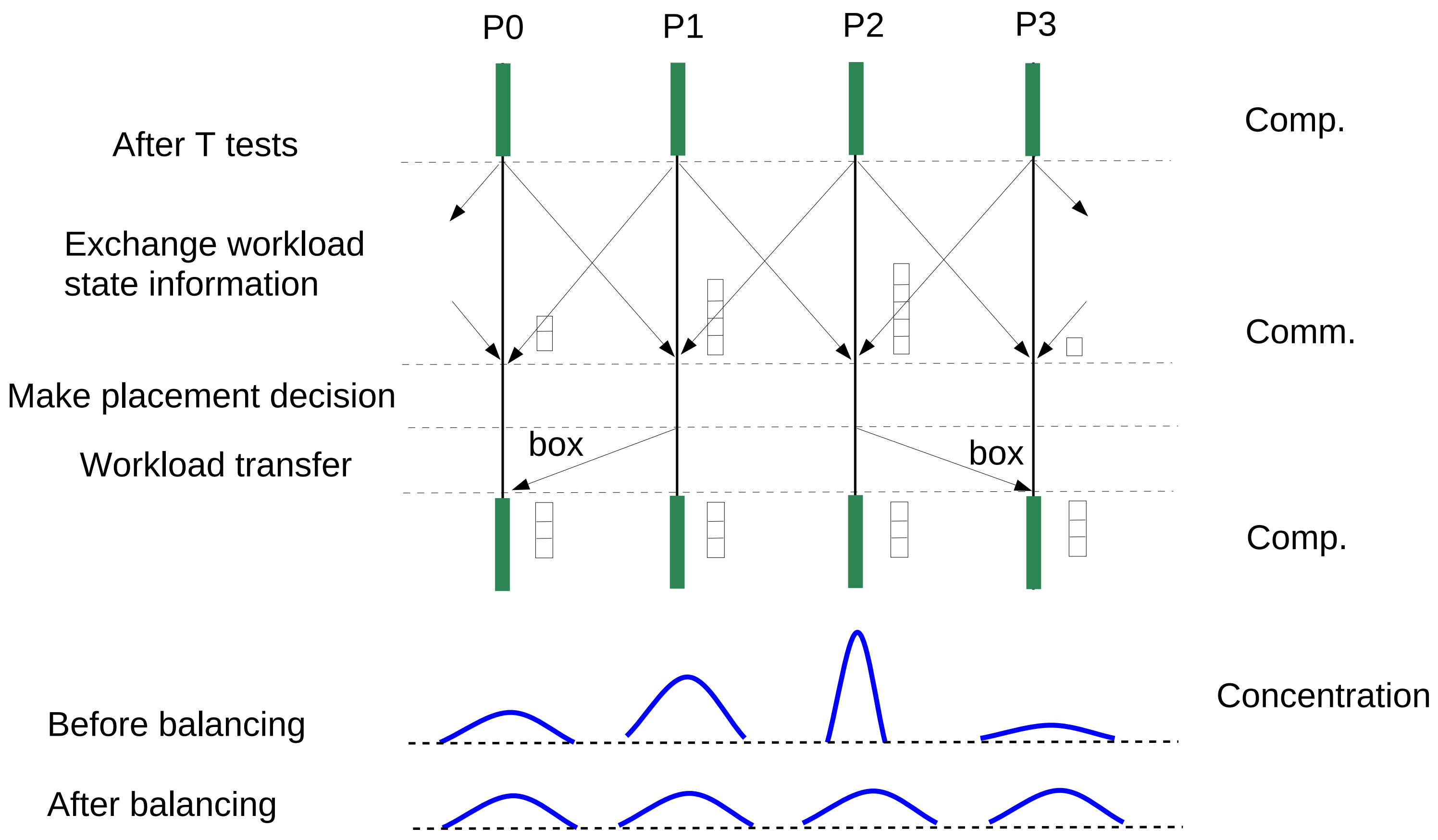
(i: local processor: j: neighbor processor)

  - If $u(j)$ is positive and greater than some tolerance: send intervals (boxes).
  - If $u(j)$ is negative and less than some tolerance: receive intervals (boxes).

- Messages have higher granularity

- Synchronism and blocking communication still cause inefficiencies.

Synchronous Diffusive Load Balancing



Asynchronous Diffusive Load Balancing

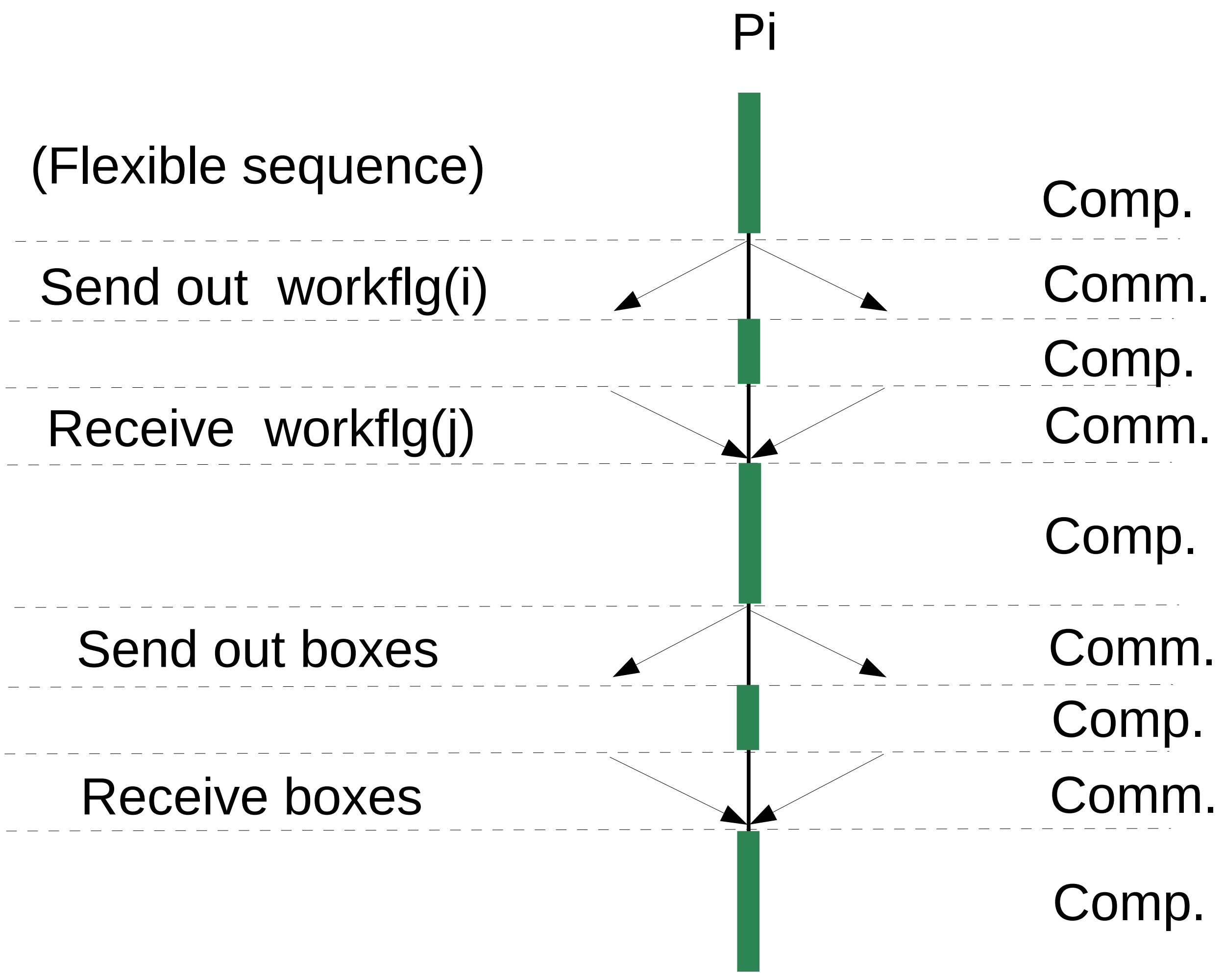
- Use asynchronous nonblocking communication to send workload information and transfer workload

- Overlaps communication and computation.

- Receiver-initiated diffusive workload transfer scheme:
  - Send out work state information only if it falls below some threshold.
  - Donor processor follows diffusive scheme to determine amount of work to send (if any).
  - Recognizes that workload balance is less important than preventing idle states.

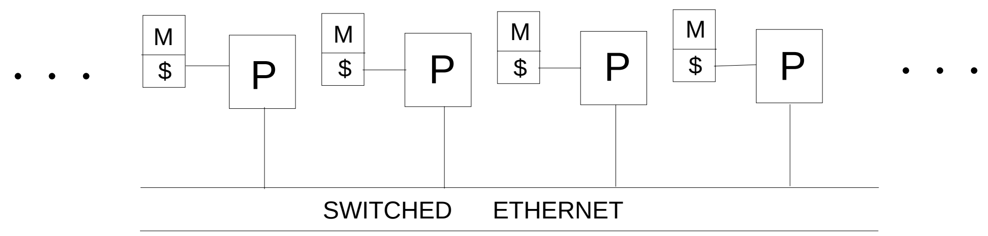
- Dijkstra’s token algorithm used to detect global termination.

Asynchronous Diffusive Load Balancing



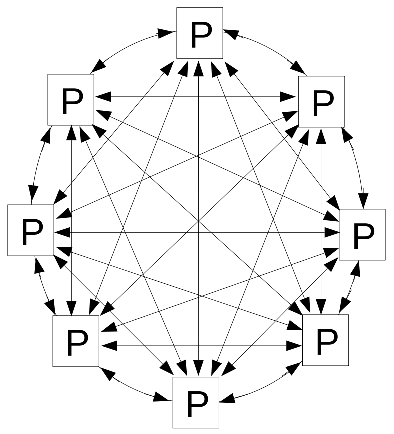
Testing Environment

- Physical hardware: Sun Ultra workstations connected by switched Ethernet (100Mbit)



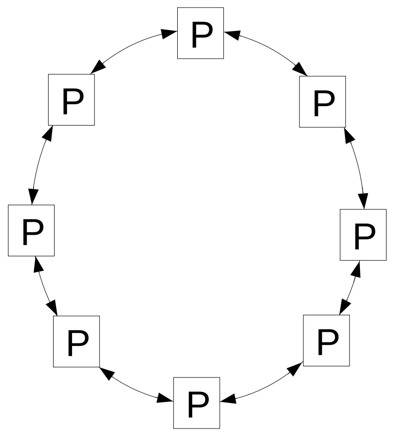
- Virtual Network:

Global Communication
All-to-All Network



Used for SWS

Local Communication
1-D Torus Network



Used for SDLB and ADLB

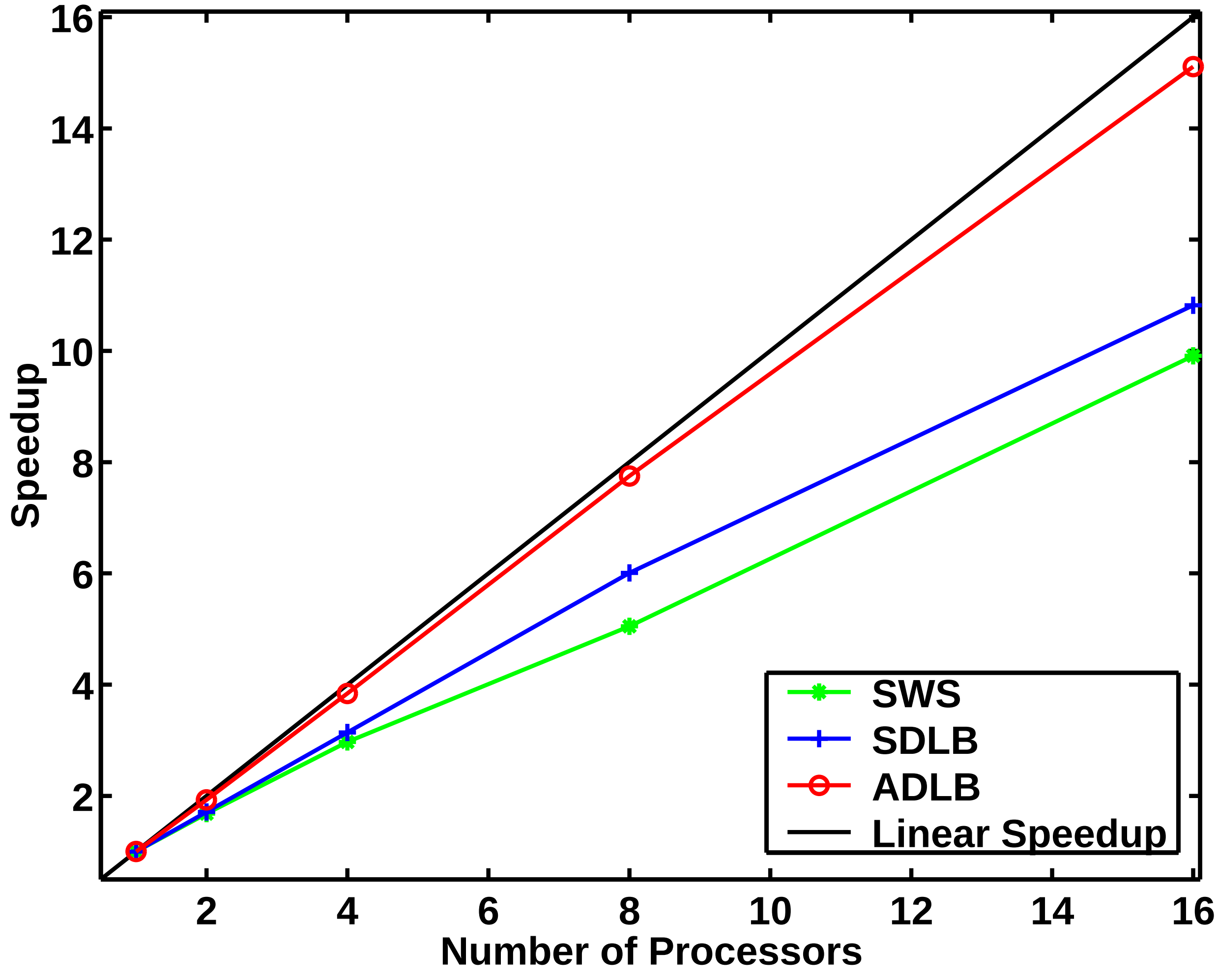
Test Problem

- Parameter estimation in a vapor-liquid equilibrium model.
- Use the maximum likelihood estimator as the objective function to determine model parameters that give the “best” fit.
- Problem data and characteristics chosen to make this a particularly difficult problem.
- Can be formulated as a nonlinear equation solving problem (which has five solutions).
- Or can be formulated as a global optimization problem.

Comparison of Algorithms on Equation-Solving Problem

Speedup vs. Number of Processors

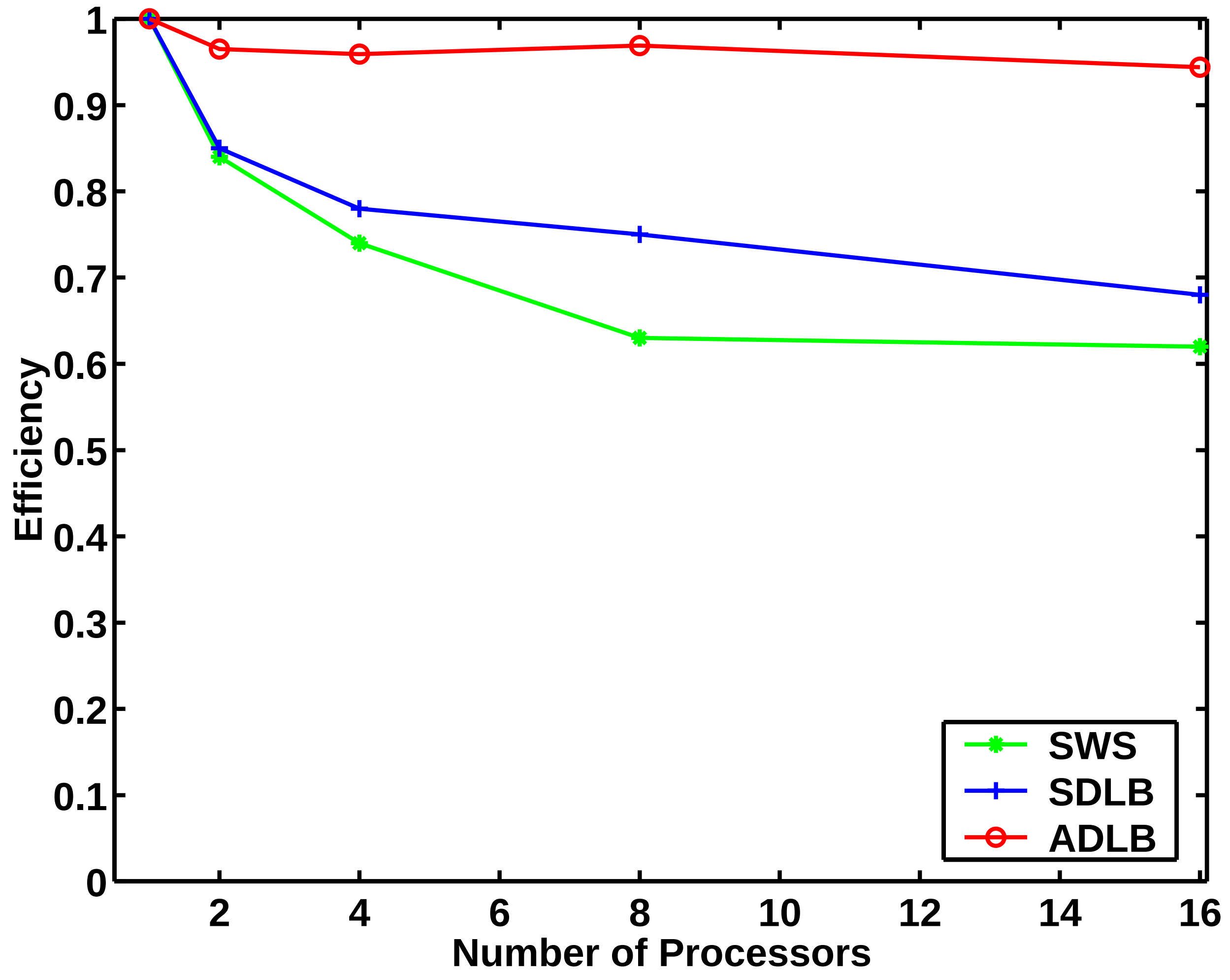
ADLB vs. SDLB vs. SWS



Comparison of Algorithms on Equation-Solving Problem

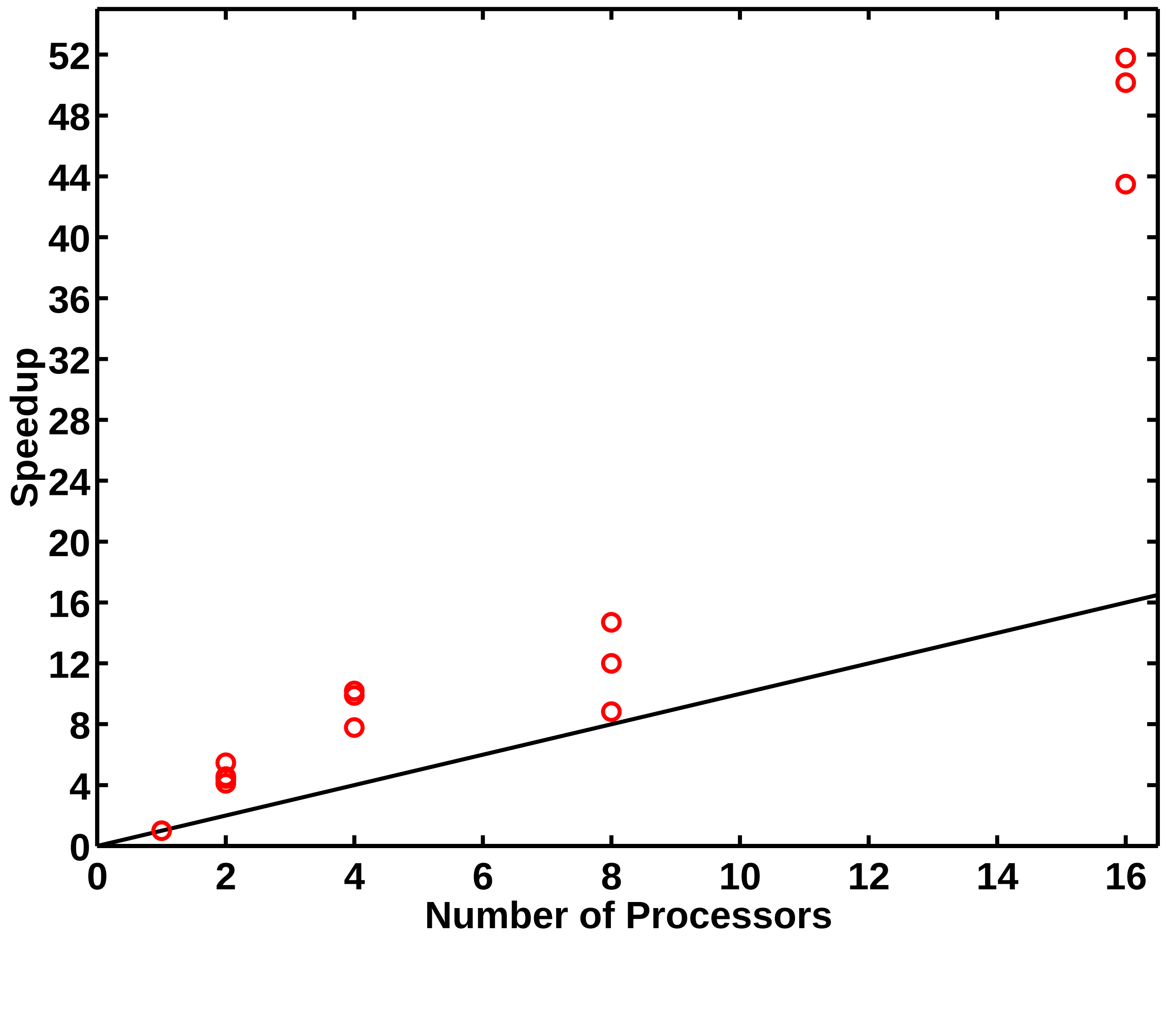
Efficiency vs. Number of Processors

ADLB vs. SDLB vs. SWS



Using ADLB on Optimization Problem

Speedup vs. Number of Processors
(three different runs of same problem)

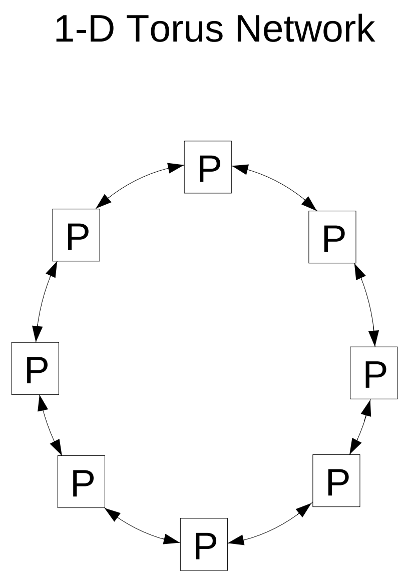


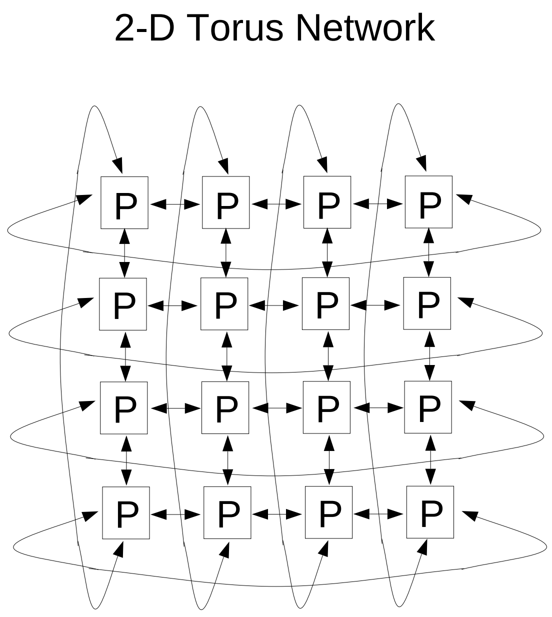
Using ADLB on Optimization Problem

- Speedups around 50 on 16 processors– superlinear speedup
- Superlinear speedup is possible because of broadcast of least upper bounds, causing intervals to do discarded earlier than in the serial case. That is, there is less work to do in the parallel case than in the serial case.
- Results vary from run to run because of different timing in finding and broadcasting improved upper bound.

Effect of Virtual Network

- We have also considered performance in a 2-D torus virtual network.





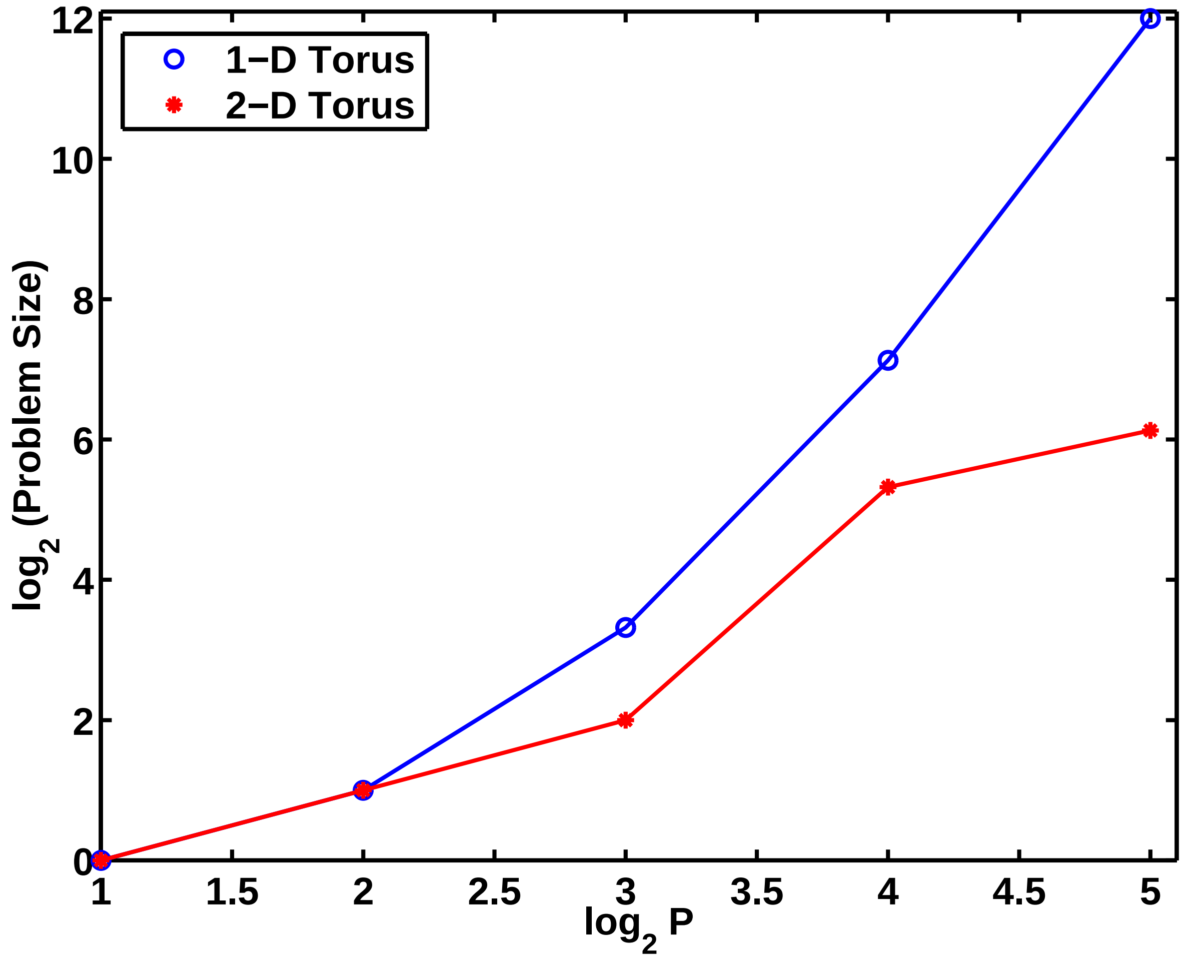
- 1-D vs. 2-D torus
  - 2-D has higher communication overhead (more neighbors)
  - 2-D has smaller network diameter (shorter message diffusion distance): $2\lfloor\sqrt{P}/2\rfloor$ vs. $\lfloor P/2\rfloor$
  - Trade off may favor 2-D for large number of processors.

Effect of Virtual Network

- ADLB algorithm was tested using both 1-D and 2-D virtual connectivity.
- The test problem is an equation solving problem: computation of critical points of mixtures.
- Comparisons made using isoefficiency analysis: As number of processors is increased, determine problem size needed to maintain constant efficiency relative to best sequential algorithm.
- Isoefficiency curves at 92% were determined up to 32 processors.

Isoefficiency Curves (92%) for Equation-Solving Problem

2-D Torus vs. 1-D Torus
(Lower is better)



Stack Management for Workload Placement

- Especially for optimization problems, the selection rule for workload transfer can have a significant effect on performance.
- With the goal of maintaining consistently high (superlinear) speedups on optimization (BB) problems, we have used a dual stack management scheme
- Each processor maintains two workload stacks, local stack and a global stack.
  - The processor draws work from the local stack in the order in which it is generated (depth-first pattern).
  - The global stack provides work for transmission to other processors.
  - The global stack is created by randomly removing boxes from the local stack, contributing breadth to the tree search process.

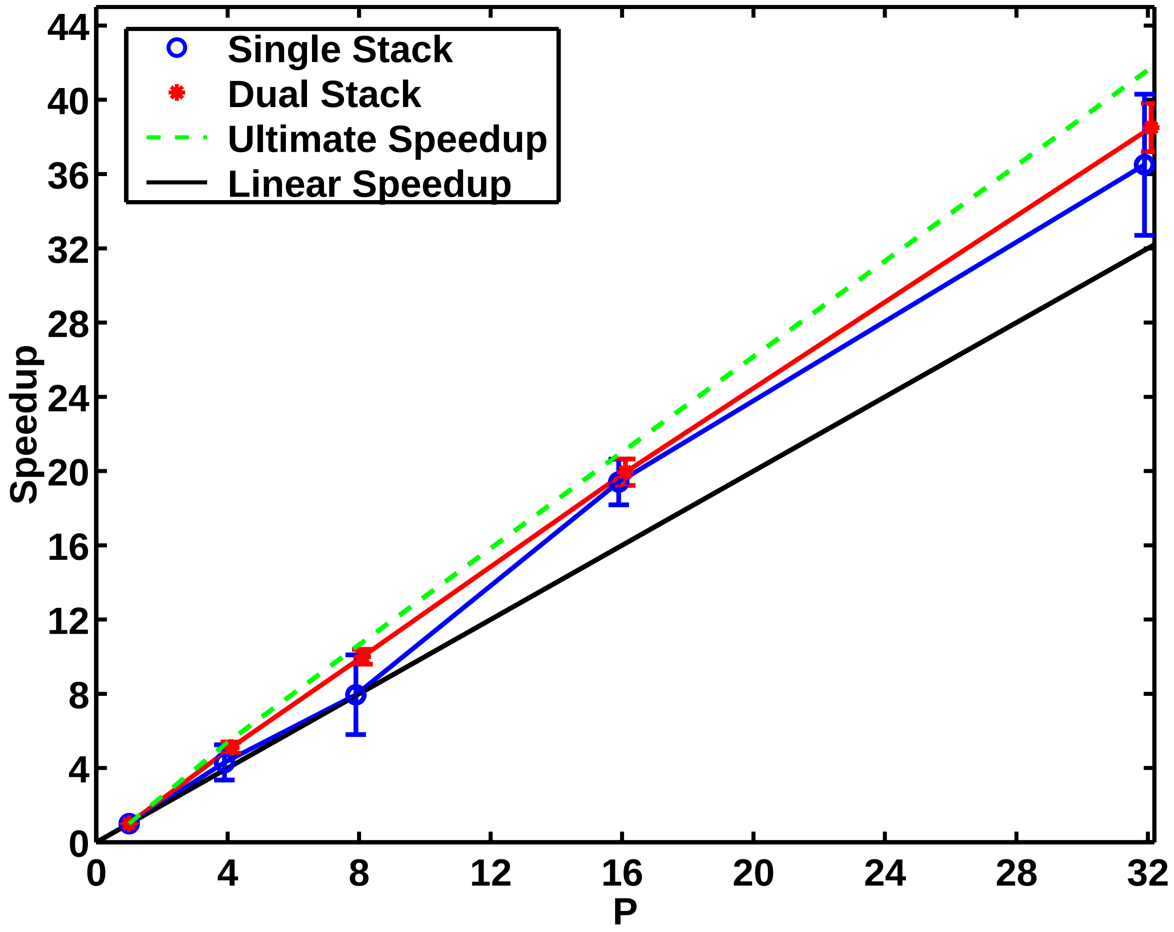
Workload Placement (cont'd)

- The dual stack strategy was tested using a 2-D torus virtual network up to 32 processors.
- The test problem was an optimization problem: parameter estimation using an error-in-variable approach.
- For comparisons, an "ultimate speedup" was determined by initially setting the best upper bound to the value of the global minimum.
- Results indicate that the dual stack strategy leads to higher speedups and less variability from run to run (based on 10 runs of each case).

Workload Placement (cont'd)

Speedup vs. Number of Processors

Dual Stack vs. Single Stack vs. Ultimate



Concluding Remarks

- Interval analysis is a powerful general-purpose and model-independent approach for solving a variety of process modeling problems, providing a mathematical and computational guarantee of reliability.

- Continuing advances in computing hardware and software (e.g., compiler support for interval arithmetic, parallel computing) will make this approach even more attractive.

- The guaranteed reliability of interval methods comes at the expense of a significant CPU requirement. Thus, there is a choice between fast local methods that are not completely reliable, or a slower method that is guaranteed to give the complete and correct answer.

- The modeler must make a decision concerning how important it is to get the correct answer.

Concluding Remarks (cont'd)

- With effective load management strategies, parallel BB and BP problems (using interval methods or other approaches) can be solved very efficiently using MPI on a networked cluster of workstations.
  - Good scalability.
  - Exploit potential for superlinear speedup in BB.

- Parallel computing technology can be used not only to solve problems faster, but to solve problems more reliably.

- These reliability issues are often overlooked:

  Are we just getting the wrong answers faster?

Acknowledgments

- American Chemical Society Petroleum Research Fund (30421-AC9)

- U. S. National Science Foundation (DMI96-96110 and EEC97-00537-CRCD)

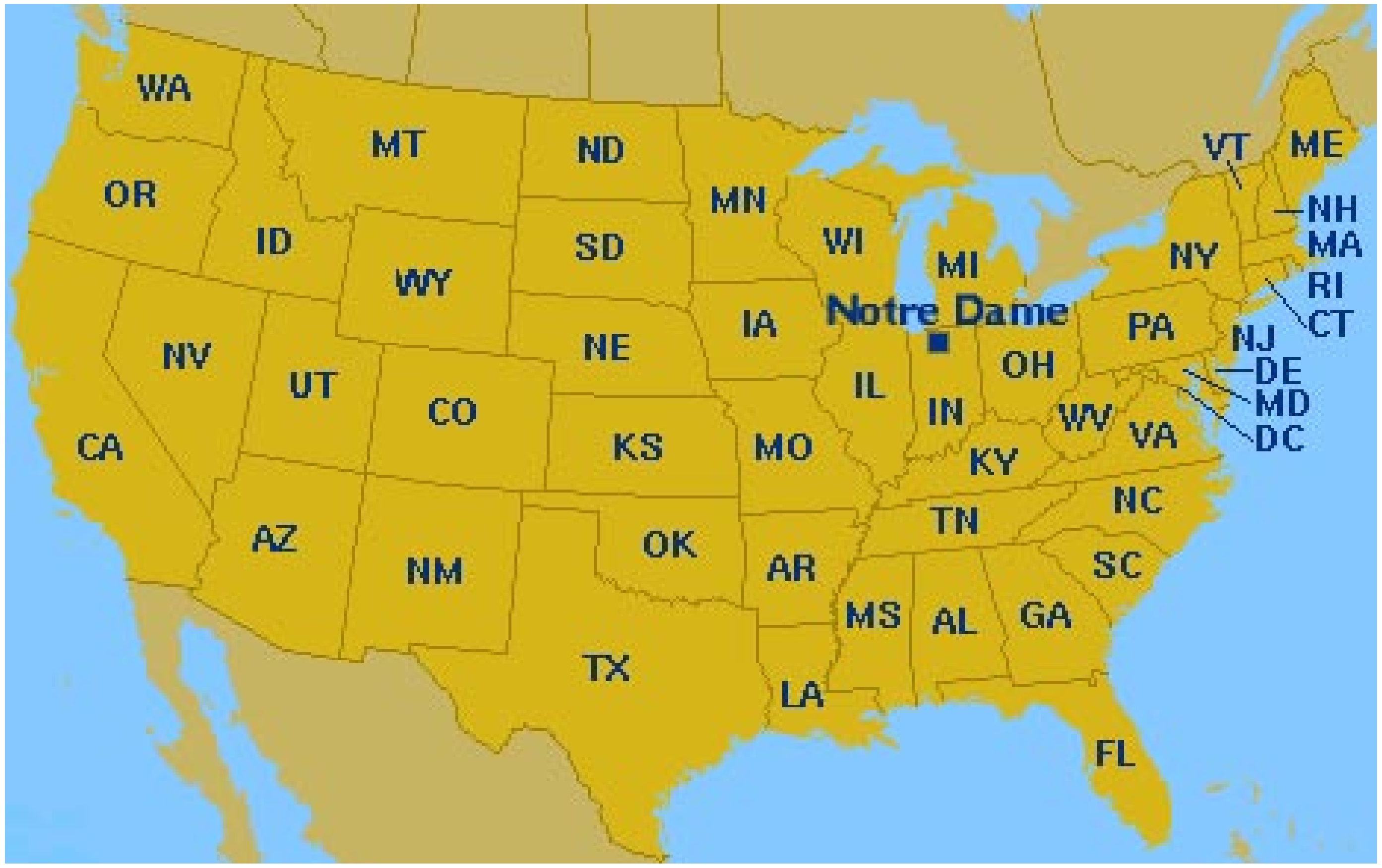
- U.S. Army Research Office (DAAG55-98-1-0091)

- Sun Microsystems, Inc.

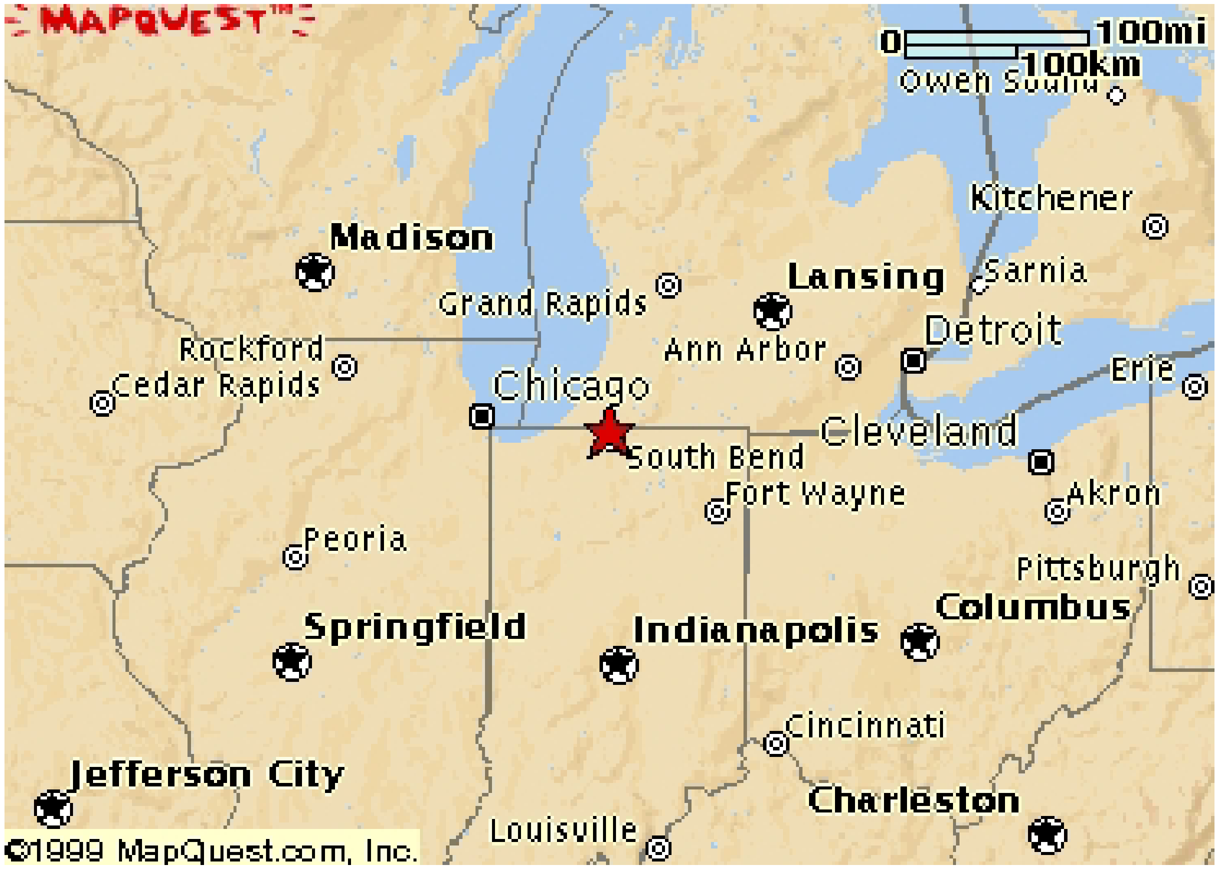
University of Notre Dame

University of Notre Dame

Where is Notre Dame?



Where is Notre Dame?



Lake Michigan

Lake Michigan