

IBM T.J. Watson Research Center

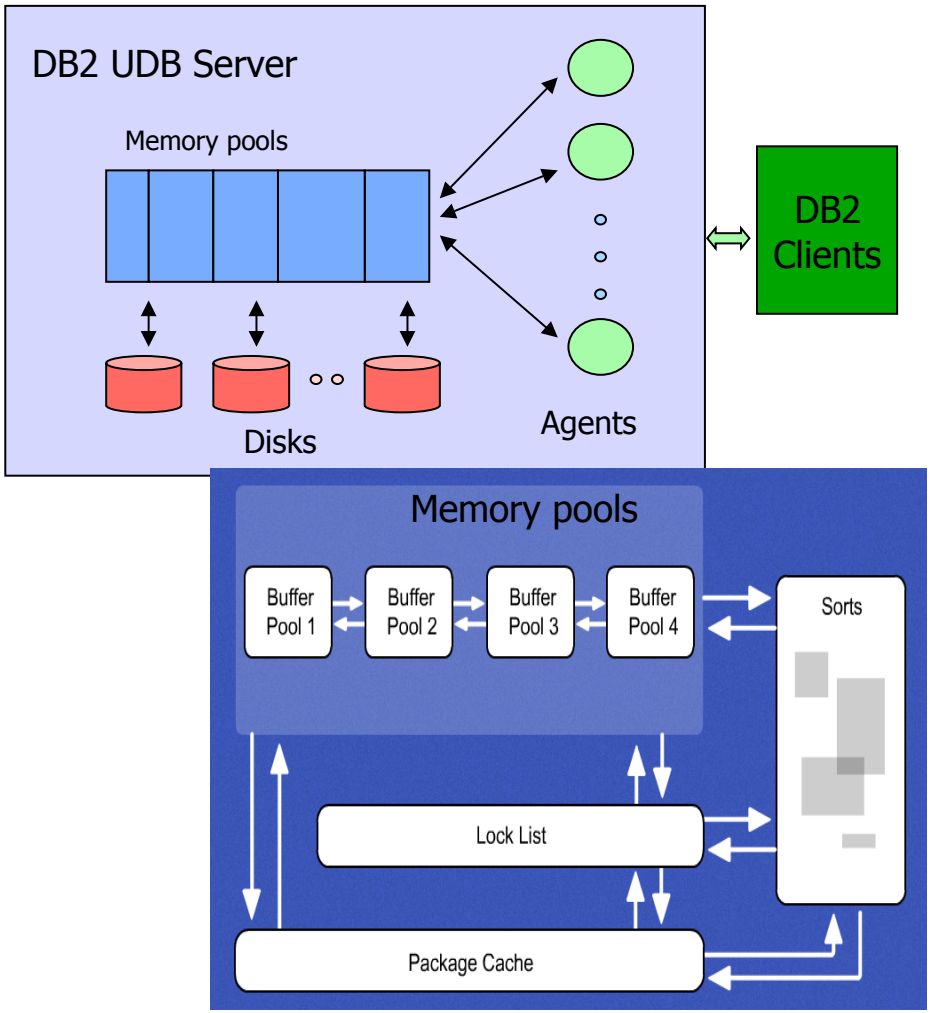
Self-Tuning Memory Management of A Database System

Yixin Diao

diao@us.ibm.com

IM 2009 Tutorial: Recent Advances in the Application of Control Theory to Network and Service Management

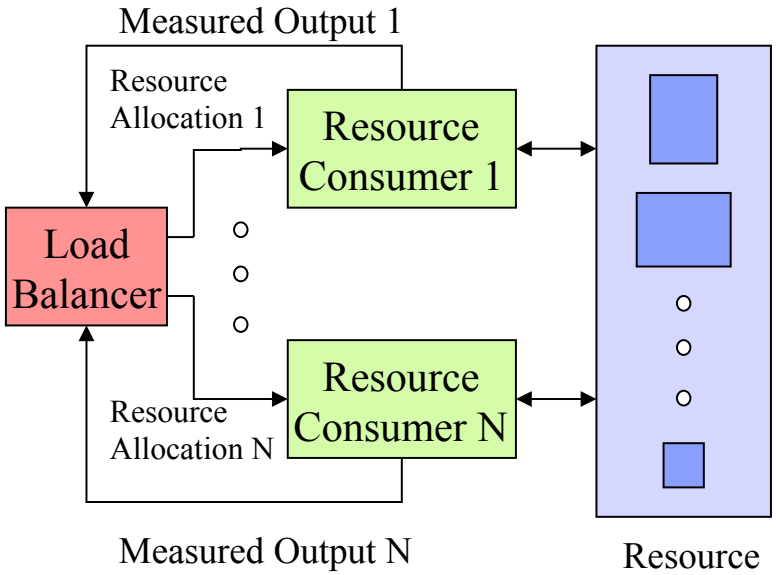
DB2 Self-Tuning Memory Management



- **Technical problems**
  - Large systems with varying workloads and many configuration parameters
  - Autonomic computing: systems self-management
- **Challenges from systems aspects**
  - Heterogeneous memory pools
  - Dissimilar usage characteristics
- **Challenges from control aspects**
  - Adaptation and self-design
  - Reliability and robustness

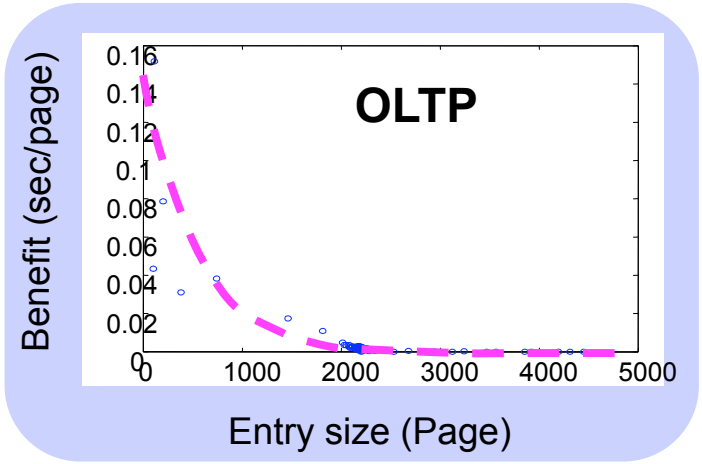
© 2009 IBM Corporation

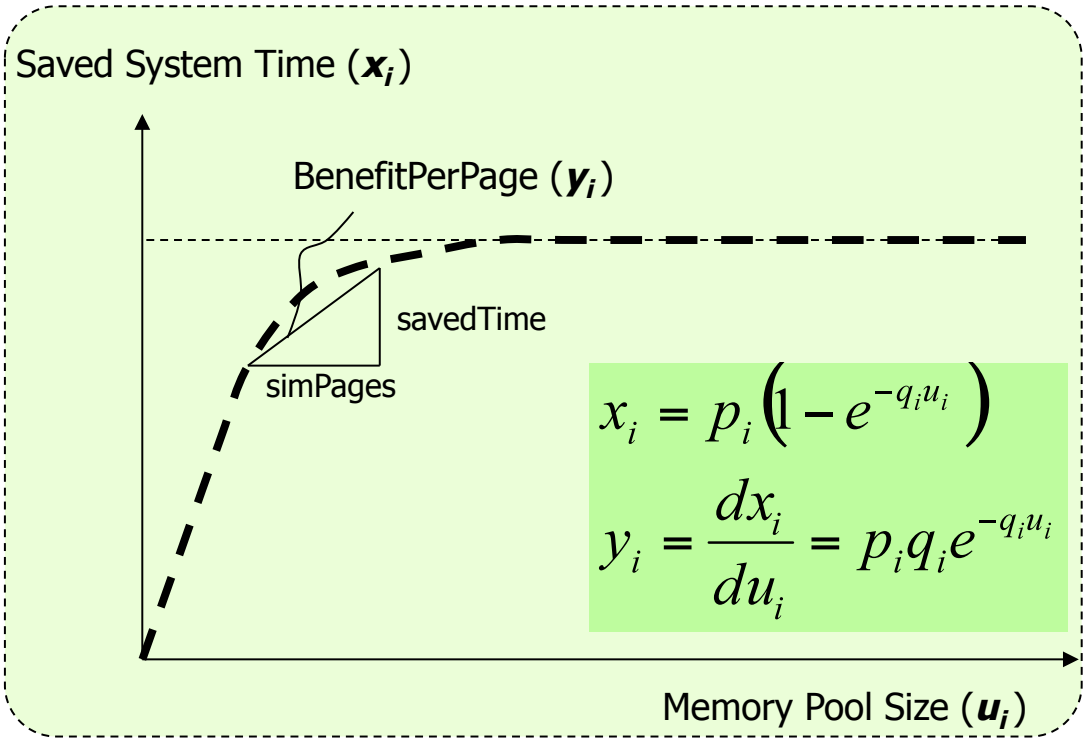
Load Balancing for Database Memory



Load Balancing

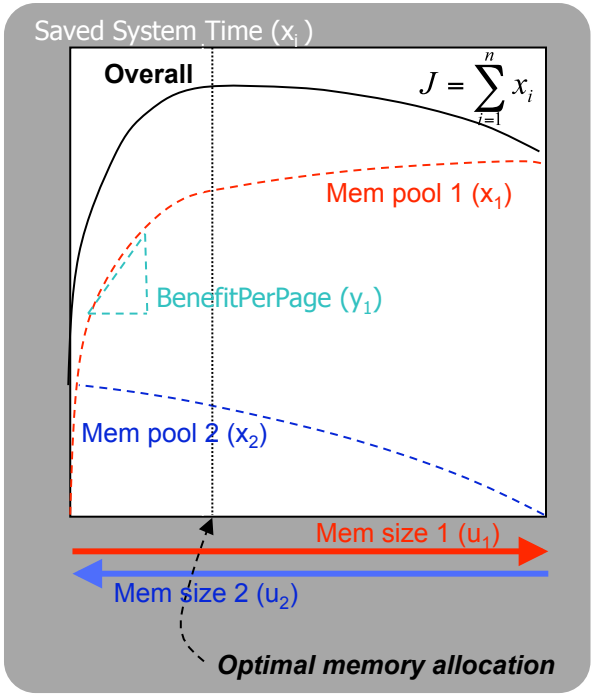
- Fairness ➔ optimal ?
- Common measured output ?

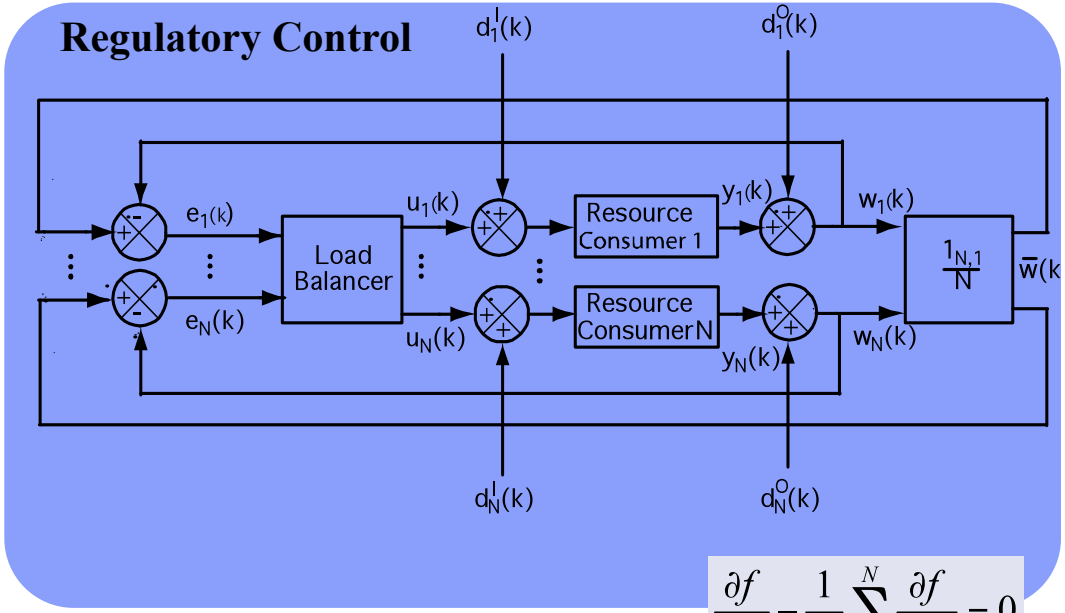




© 2009 IBM Corporation

Constrained Optimization and Regulatory Control





$$\frac{\partial f}{\partial u_i} - \frac{1}{N}\sum_{j=1}^{N}\frac{\partial f}{\partial u_j} = 0$$

Constrained Optimization

$$J = f(u_1, u_2, \ldots, u_N)$$

$$g(u_1, u_2, \ldots, u_N) = \sum_{i=1}^{N} u_i - U = 0$$

$$h(u_1, u_2, \ldots, u_N) = u_i - b_i \geq 0$$

Karush-Kuhn-Tucker conditions

$$L = f(u_1, u_2, \ldots, u_N) + \lambda g(u_1, u_2, \ldots, u_N) + \mu' h(u_1, u_2, \ldots, u_N)$$

$$\frac{\partial L}{\partial u_i} = \frac{\partial f}{\partial u_i} + \lambda + \mu_i = 0$$

$$\mu_i = 0 \text{ if } u_i > b_i; \quad \mu_i > 0 \text{ if } u_i = b_i$$

© 2009 IBM Corporation

Dynamic State Feedback Controller

- State space model

$$y(k+1) = Ay(k) + B\left(u(k) + d^{I}(k)\right)$$

- Control error

$$e(k) = \left(\frac{1}{N}1_{N,N} - I\right)\left(y(k) + d^{O}(k)\right)$$

- Integral control error

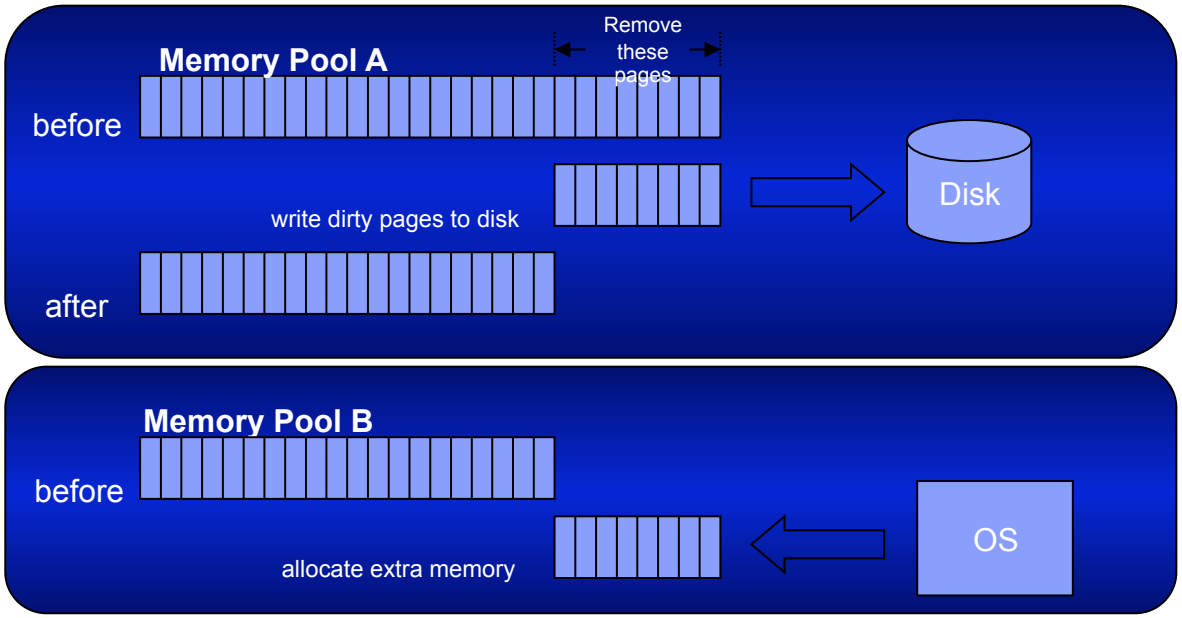
$$e_{I}(k+1) = e_{I}(k) + e(k)$$

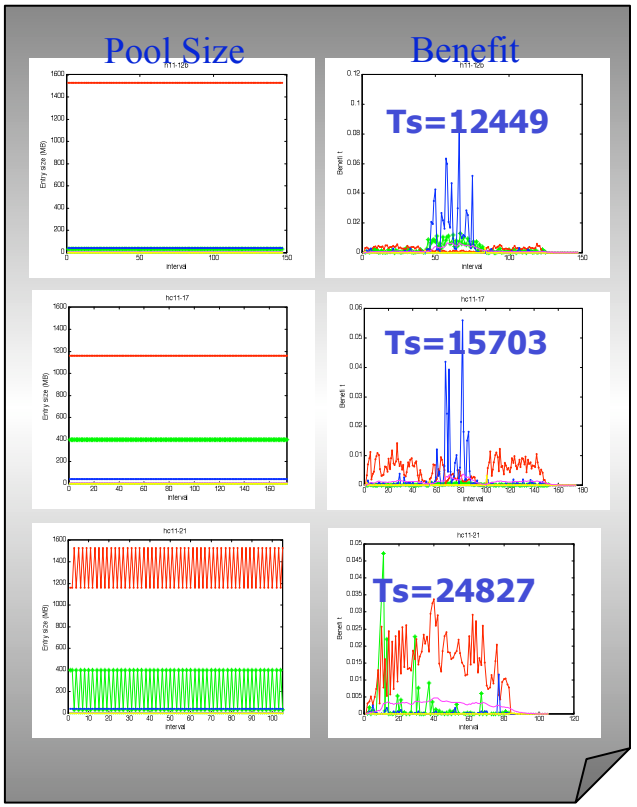
- Feedback control law

$$u(k) = K_{P}e(k) + K_{I}e_{I}(k)$$

© 2009 IBM Corporation

Incorporating Const of Control into Controller Design





Major cost: write dirty, move memory, victimize hot

Linear quadratic regulation (LQR)

$$J = \int [e^T(k)\ e^T_I(k)]\ Q\ [e^T(k)\ e^T_I(k)]^T + u^T(k)\ R\ u(k)$$

Define Q and R regarding to performance

- Cost of transient load imbalances
- Cost of changing resource allocations

© 2009 IBM Corporation

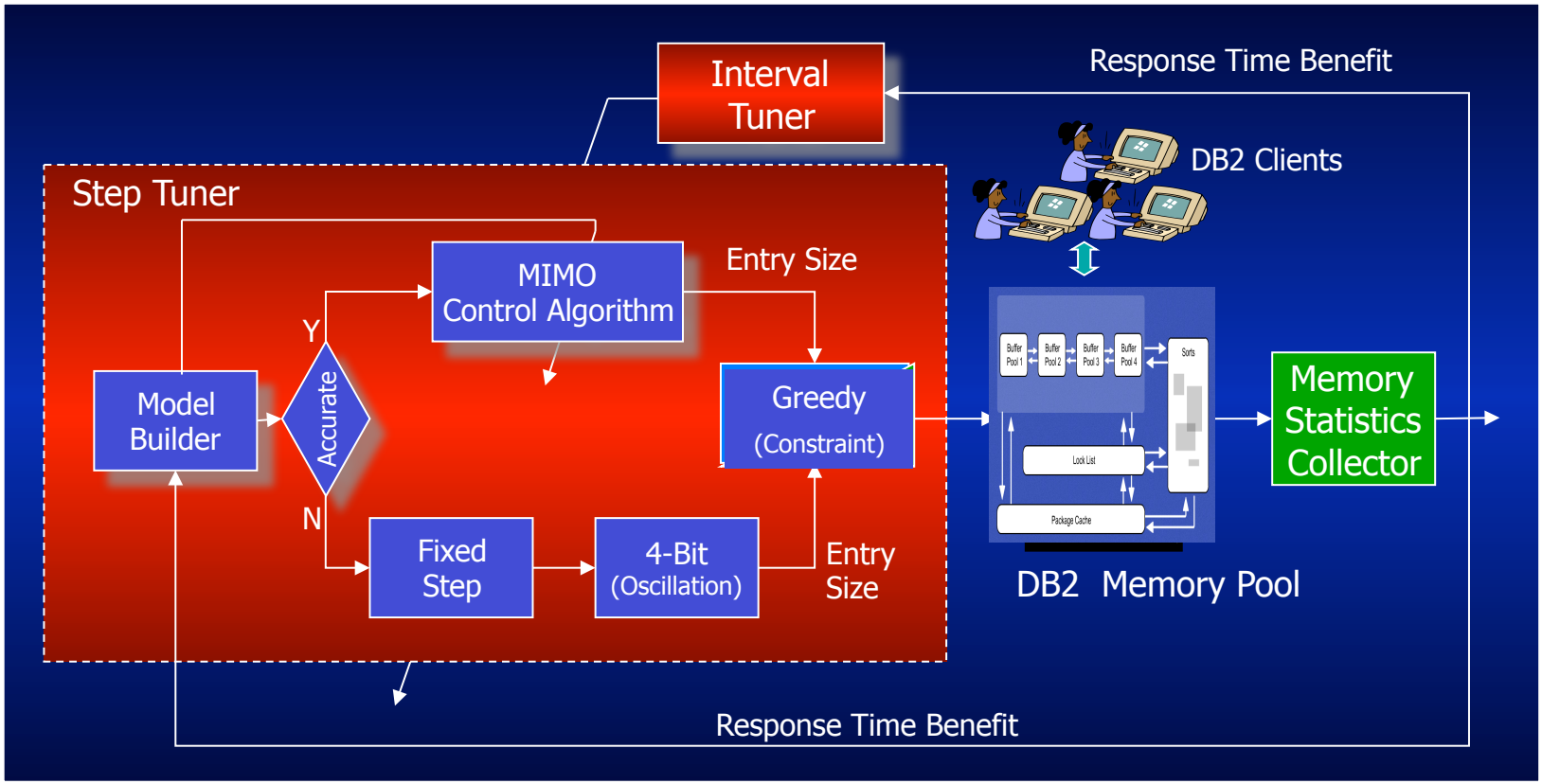
Adaptive Controller Design

Local linear model

$$y_i(k+1) = b_i(k)u_i(k)$$

Decentralized integral control

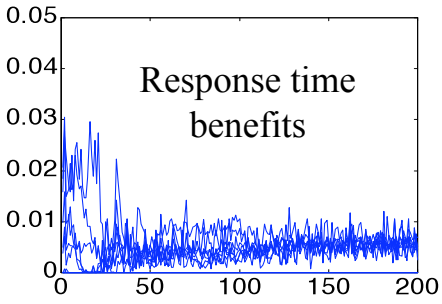
$$u_i(k+1) = u_i(k) - \frac{1-p}{b_i(k)}\left(y_i(k) - \frac{1}{N}\sum_{j=1}^{N} y_j(k)\right)$$

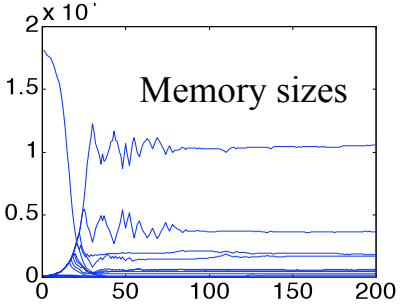


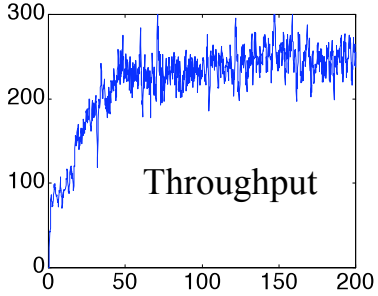
© 2009 IBM Corporation

Experimental Assessment

- **OLTP workload: multiple (20) buffer pools**

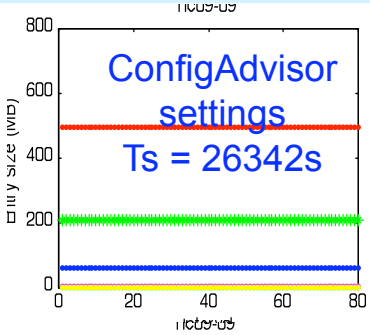


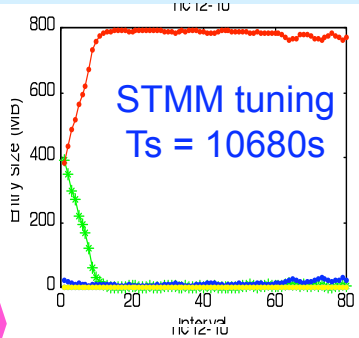


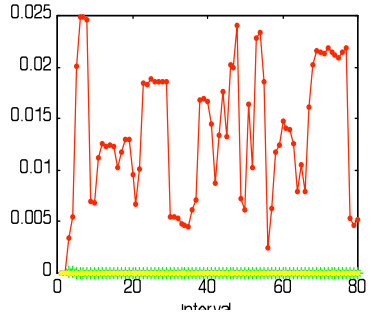


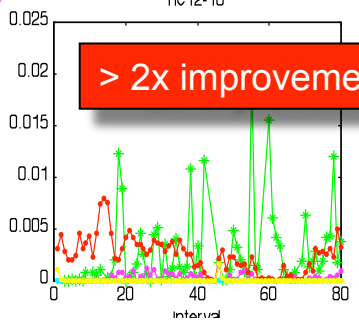
Increase TP from ~100 to ~250

- **DSS workload: various query lengths**



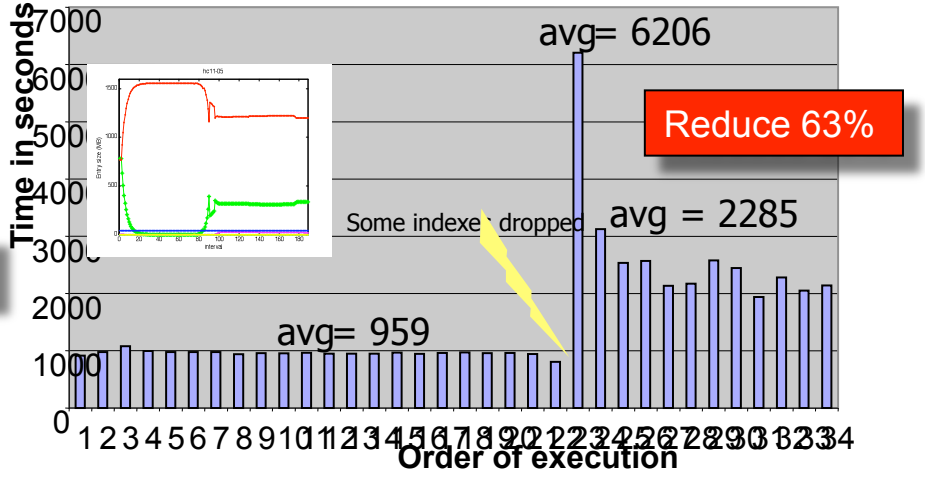






> 2x improvement

- **DSS workload: index drop**



© 2009 IBM Corporation

Comparing Control and Optimization Techniques

Control-based approach

Local linear model

$$y_i(k+1) = b_i(k)u_i(k)$$

Decentralized integral control

$$u_i(k+1) = u_i(k) - \frac{1-p}{b_i(k)}\left(y_i(k) - \frac{1}{N}\sum_{j=1}^{N} y_j(k)\right)$$

Constraint enforcement (projection method)

$$d_i(k) = \frac{u_i(k)}{\sum_{j=1}^{N} u_j(k)}\left(U - \sum_{j=1}^{N} \underline{u}_j\right) + \underline{u}_i$$

Less dependence on the model

Optimization-based approach

Gradient method

$$u(k+1) = u(k) + \lambda(k)p(k)$$

Projected gradient (quasi-Newton)

$$p(k) = -(H(k) - H(k)A^{\top}(k)\left(A(k)H(k)A^{\top}(k)\right)^{-1} A(k)H(k))y(k) \quad (14)$$

Step length (modified Armijo rule)

$$\nabla f\left(u(k) + \lambda(k)p(k)\right)^{\top} p(k) \geq c_1 \nabla f\left(u(k)\right)^{\top} p(k)$$

Strictly applies constrained optimization

Similarity in a simplified scenario

$$A(k) = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}$$

$$H(k) = \begin{bmatrix} \frac{1}{s_1(k)} & 0 & \cdots & 0 \\ 0 & \frac{1}{s_2(k)} & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \cdots & \frac{1}{s_N(k)} \end{bmatrix}$$

$$u_i(k+1) = u_i(k) - \frac{\lambda(k)}{s_i(k)}\left(y_i(k) - \frac{1}{\sum_{j=1}^{N}\frac{1}{s_j(k)}}\sum_{j=1}^{N}\frac{y_j(k)}{s_j(k)}\right)$$

Differences in design considerations

- **"Pure" average vs. convex sum**
- **Pole location vs. Armijo rule**
- **Steady-state gain vs. Hessian matrix**

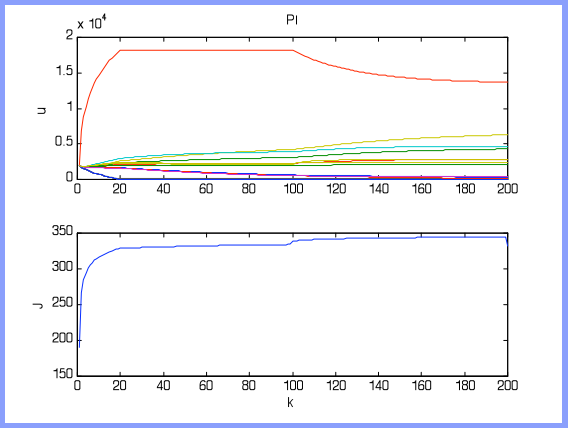
© 2009 IBM Corporation

Simulation Study: Comparison with Optimization Approach

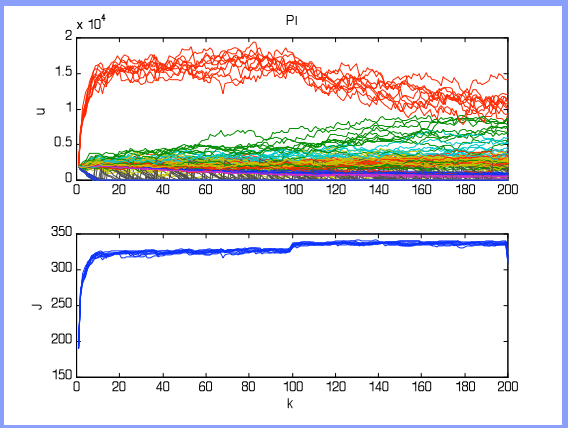
Control-based approach

Memory size

Total saved time



Control intervals

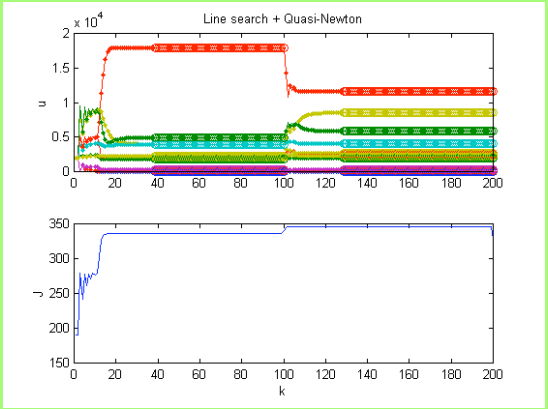


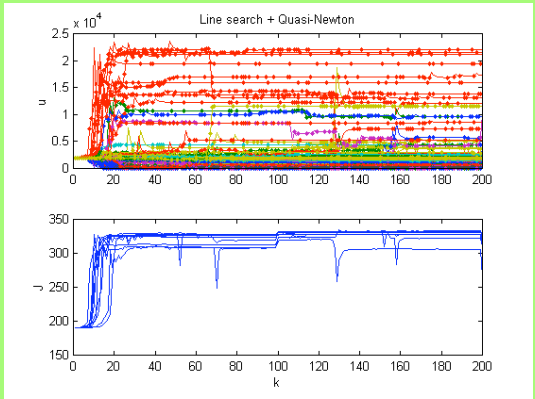
Without noise (single run)

Effect of noise (multiple runs)

More robust and better uncertainty management

Optimization-based approach





Faster convergence, but more sensitive to noise

© 2009 IBM Corporation

Summary

- **DB2 self-tuning memory management**
  - Interconnection, heterogeneity, adaptation and robustness, cost of control
  - Constrained optimization with a linear feedback controller
  - Experimental assessment for OLTP and DSS workloads

© 2009 IBM Corporation