

次世代オペレーティングシステム

SSS-PC

The Scalable Operating System



Jan 20, 2003

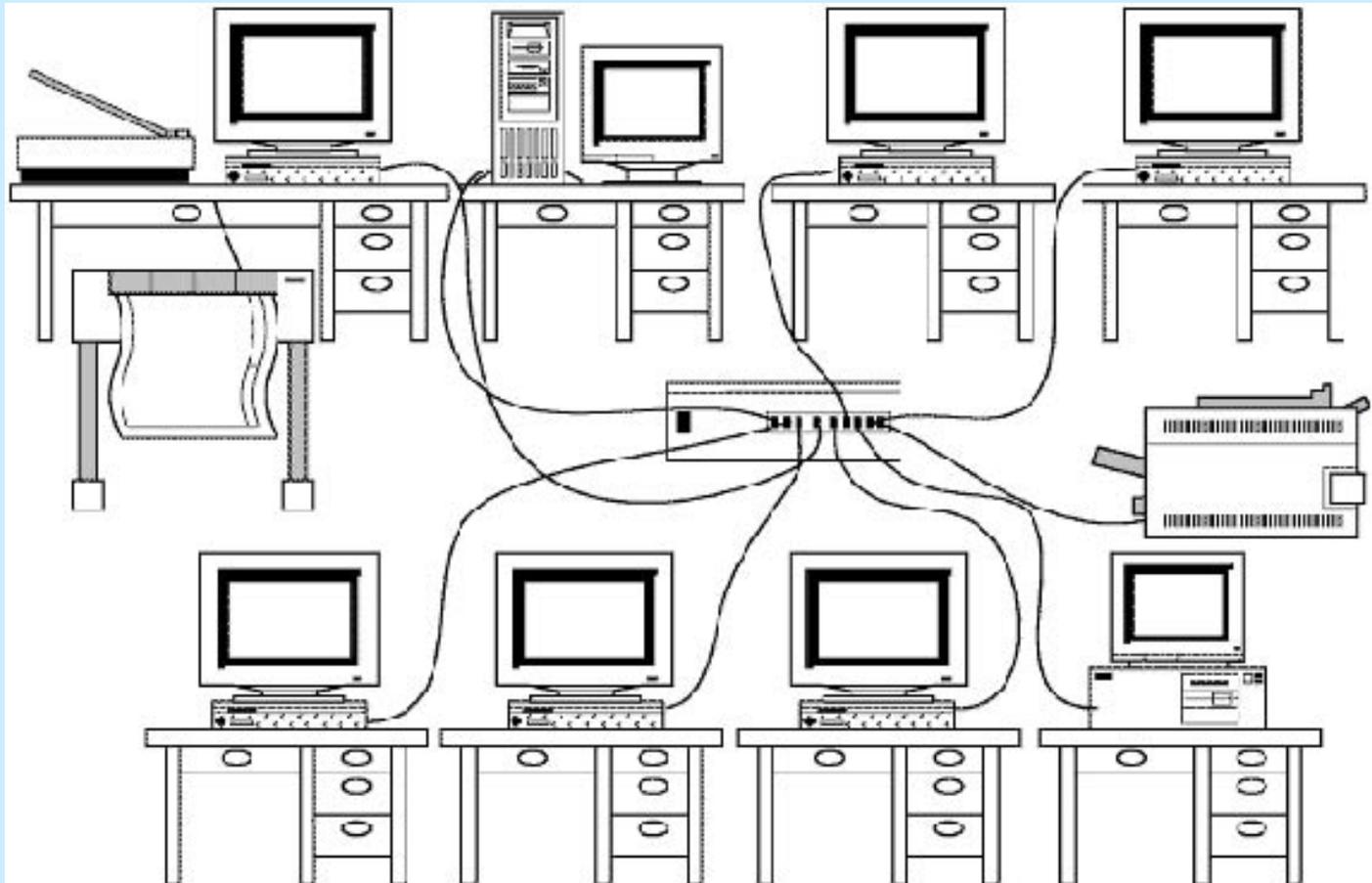


松本 尚

国立情報学研究所
情報基盤研究系

対象となるクラスタシステム(1)

職場におけるクラスタ(計算機群)



対象となるクラスタシステム(2)

PCサーバークラスタ



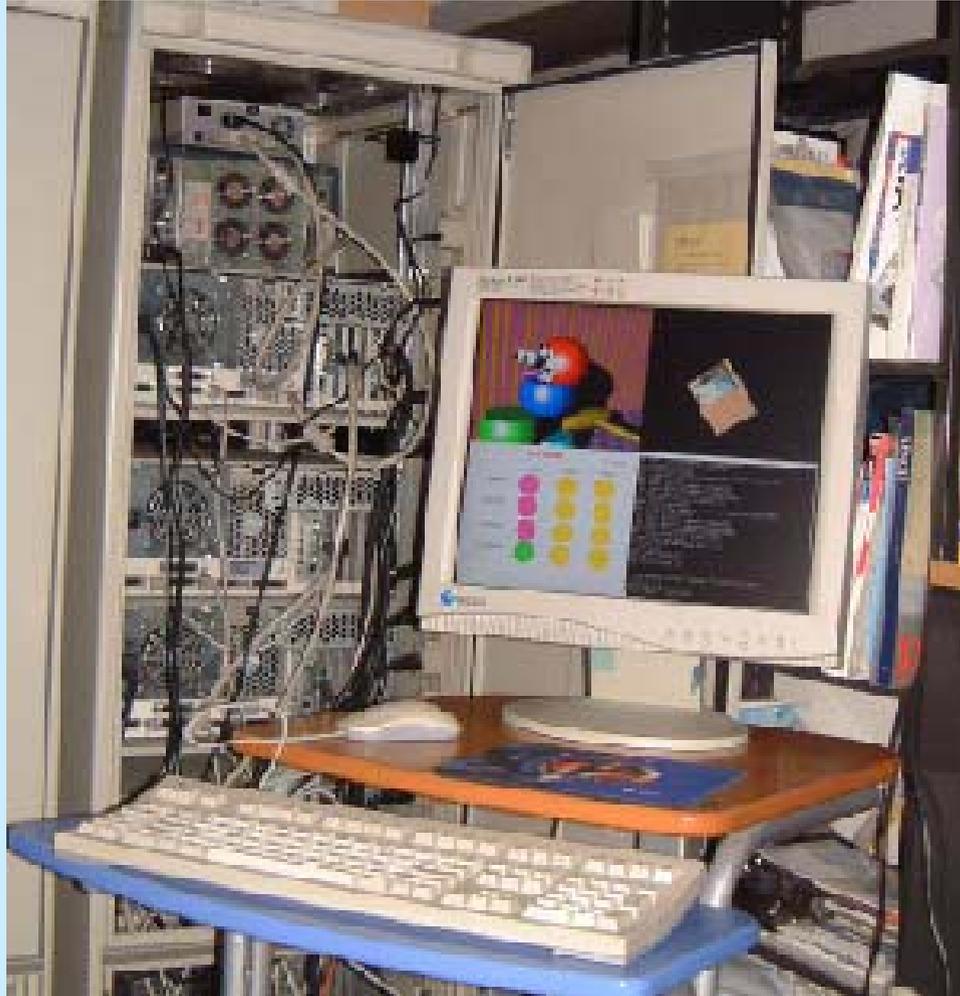
対象となるクラスタシステム(3)

PC & WS 混載クラスタ



対象となるクラスタシステム(4)

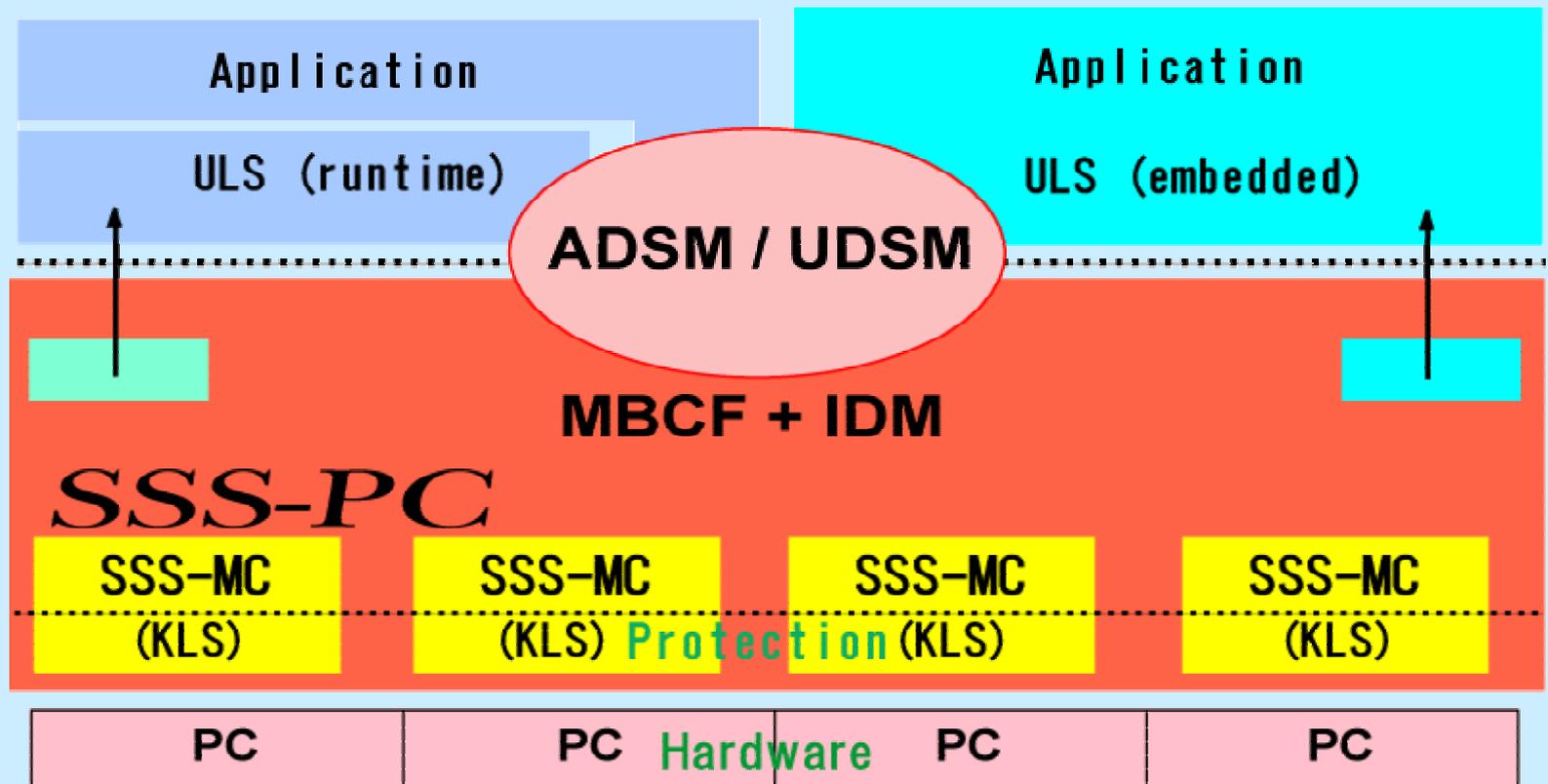
SSS-PC WSクラスタ



SSS PC の特徴

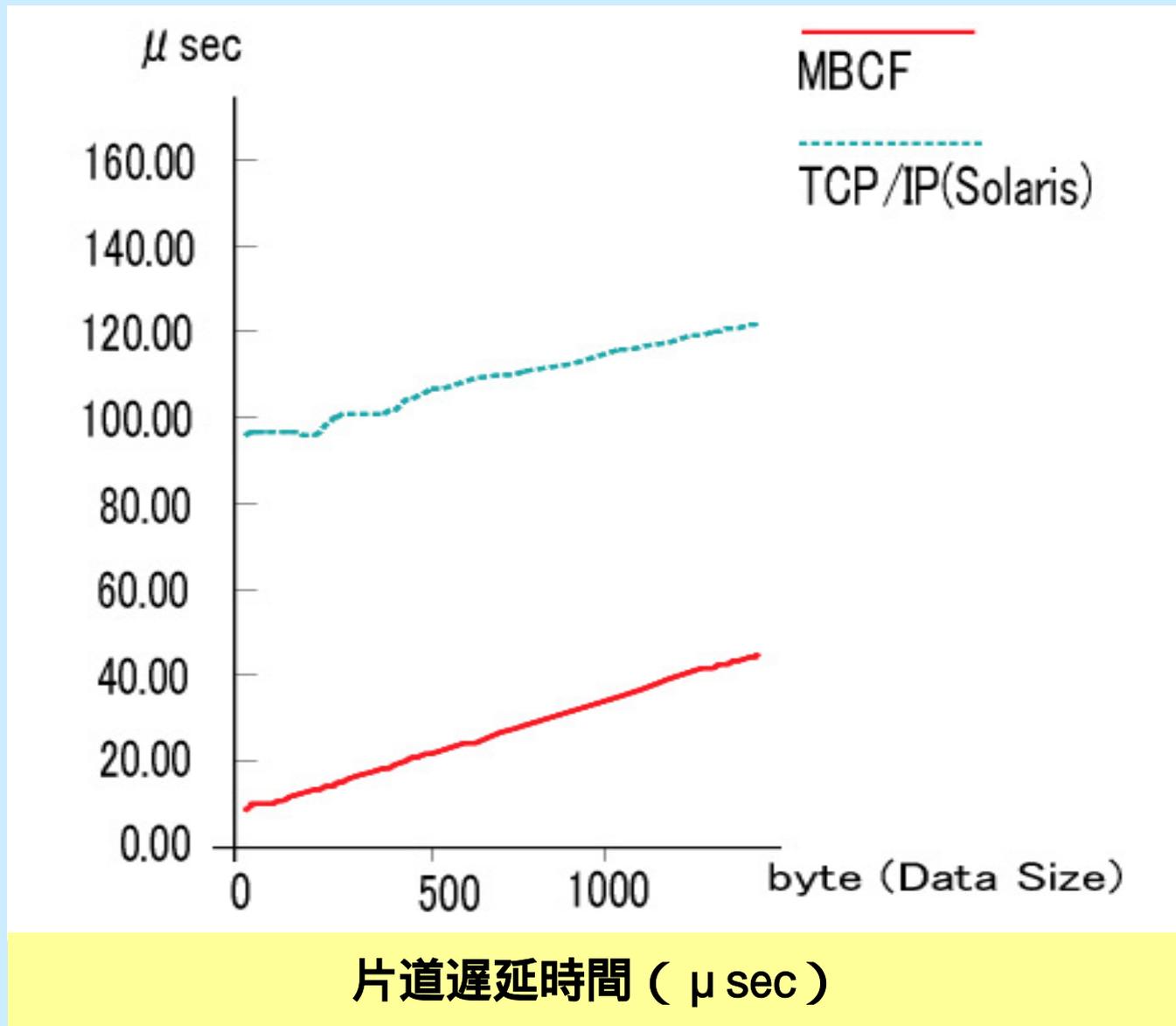
1. 高機能、低遅延、低オーバーヘッドの汎用通信方式: MBCF
2. タスクマイグレーション機能による無停止メンテナンス
3. 自由市場原理(FMM)に基づく負荷分散機能
4. マシン境界を越えた情報開示機構: IDM
5. LINUX・UNIXプログラムとのソースコード互換開発環境
6. 分散共有メモリサポート: ADSM/UDSM
7. 並列プログラムの多重並行処理(マルチタスク)
8. 共有メモリプログラム用最適化コンパイラ: RCOP
9. 高性能MPIライブラリ: MPI/MBCF
10. 純国産オペレーティングシステム

SSS PC Ver. 1.0 の機能構成

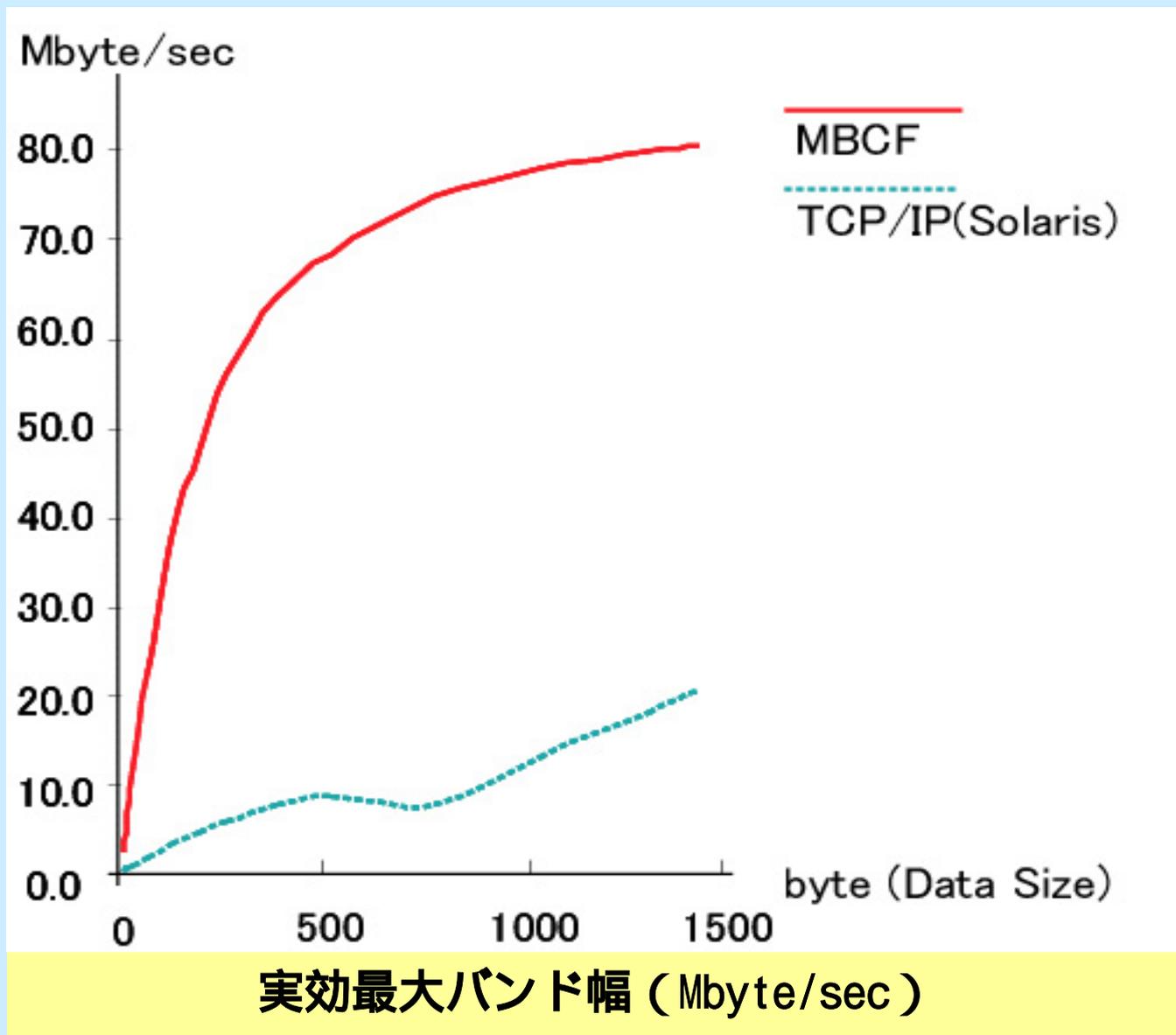


MBCF: Memory-Based Communication Facilities
 ADASM: Asymmetric Distributed Shared Memory
 UDSM: User-level Distributed Shared Memory
 KLS: Kernel Level Scheduler
 ULS: User Level Scheduler
 IDM: Information Disclosure Mechanism

Gigabit Ethernetを使った MBCFの性能(1)



Gigabit Ethernetを使った MBCFの性能(2)

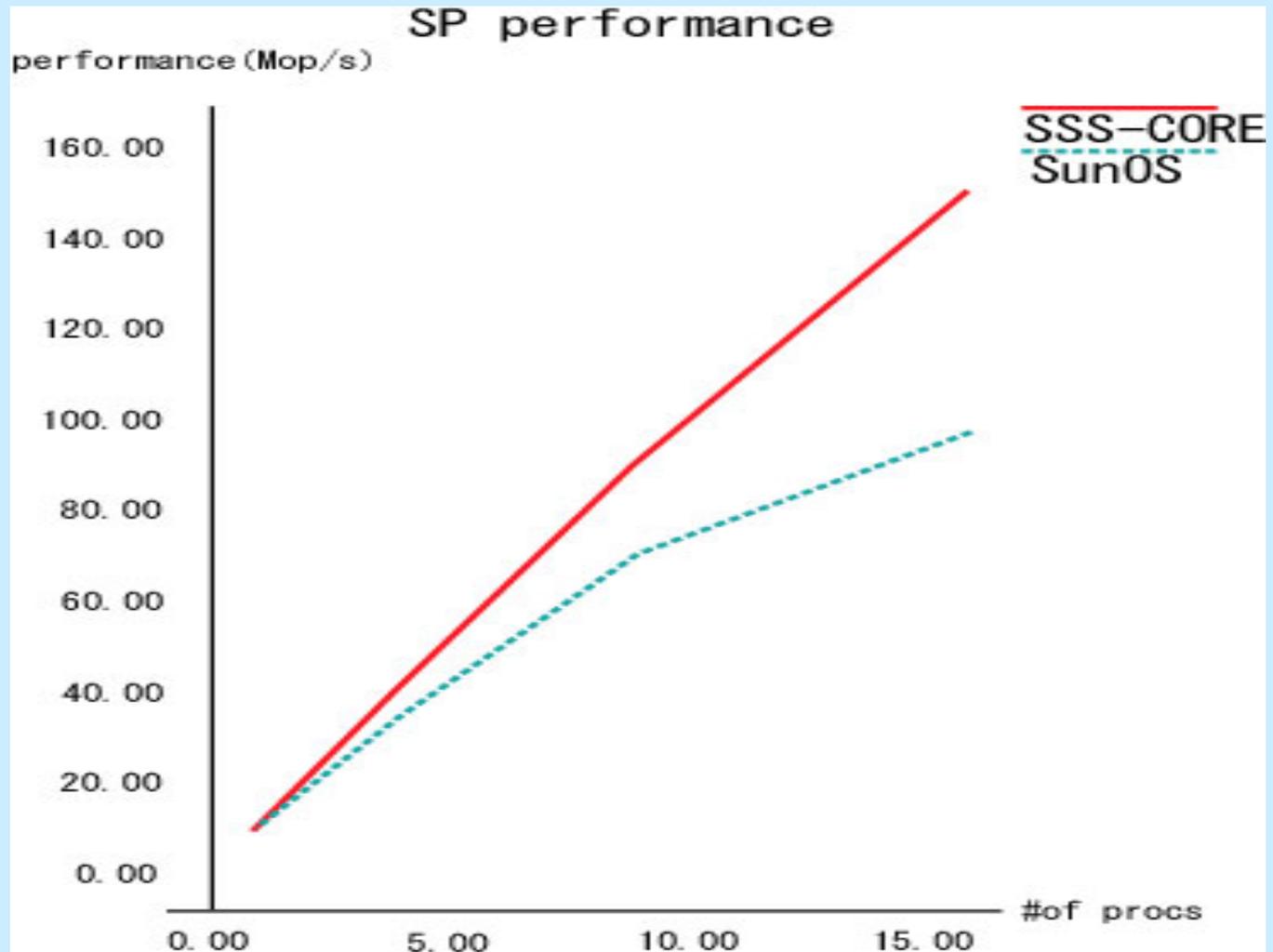


MPI/MBCF@SSS-PC と MPICH@SunOSの性能比較 (1)



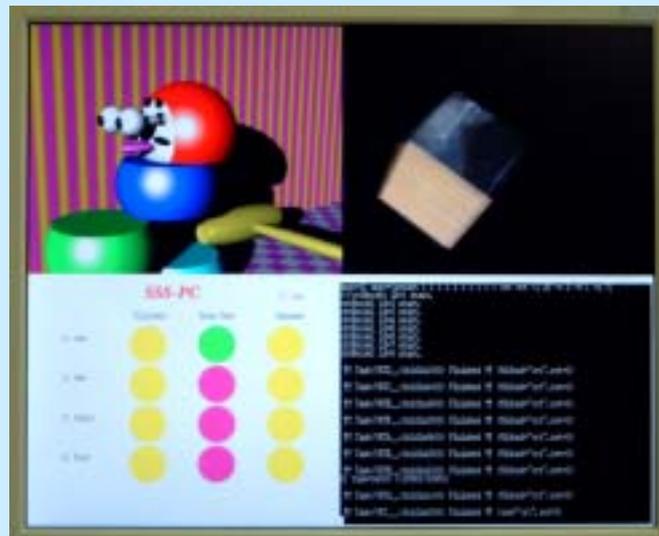
SSS-PC におけるアプリケーション実行性能
(SunOSにおける性能を1とした。16ノード。
性能 = 実行時間の逆数)

MPI/MBCF@SSS-PC と MPICH@SunOSの性能比較 (2)

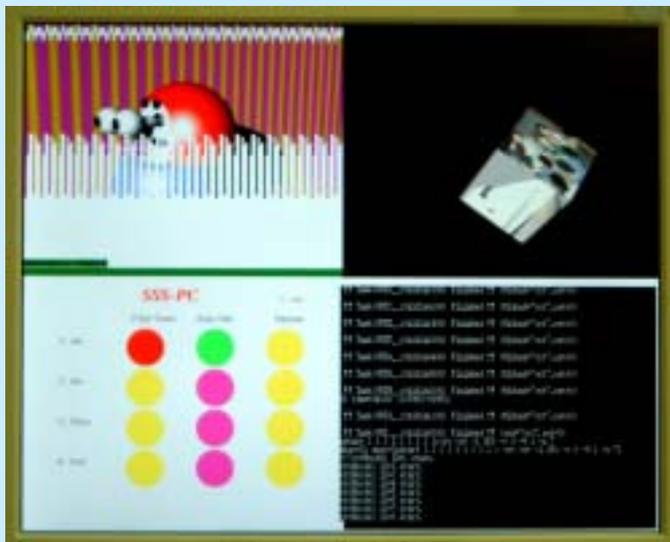


SP実行時の台数効果

タスクマイグレーション(仕事移送)機能 1



1番で仕事1個、2,3,4番仕事各3個を固定



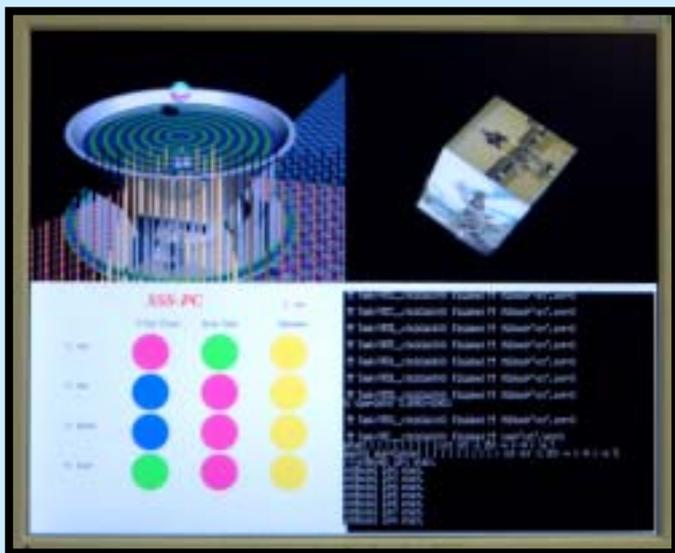
1番で仕事8個を起動



8個の仕事(左上)が他のマシンに移動中



タスクマイグレーション(仕事移送)機能 2



準安定状態



右上のアプリ停止後の安定状態(各2個の仕事)



2番に4個の固定仕事を新規投入

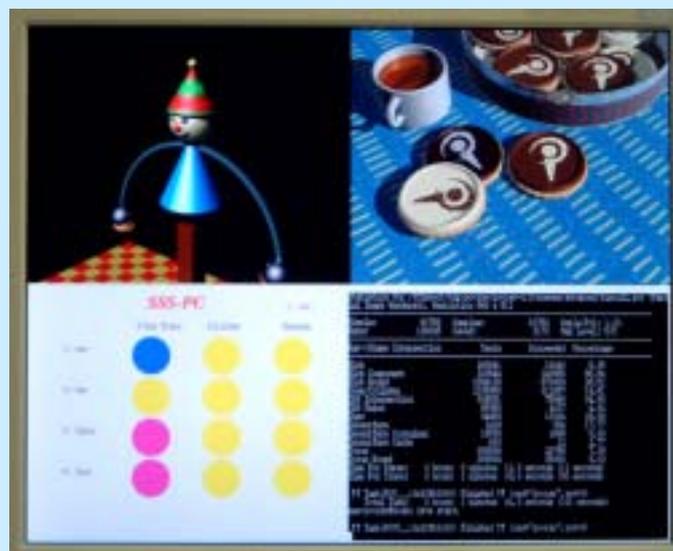


左上のアプリは2番から3,4番に退避

タスクマイグレーション(仕事移送)機能 3



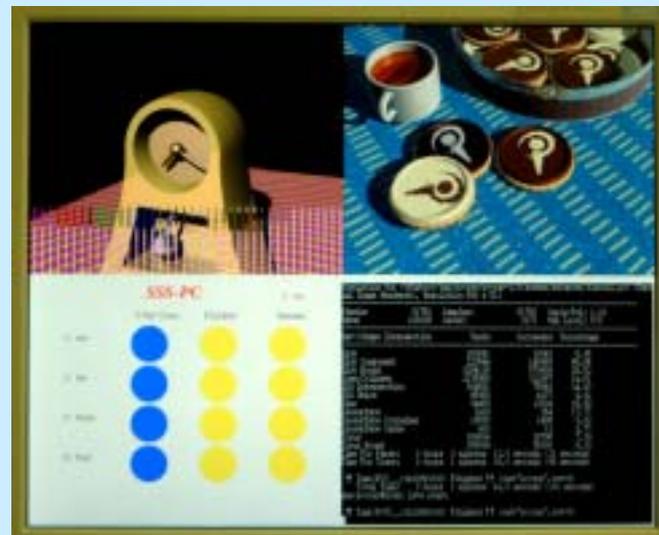
右上の新規投入アプリ実行中(定常状態)



右上の新規アプリが終了



空いた2番に仕事が移動



再び仕事2個ずつの定常状態に

タスクマイグレーション(仕事移送)機能 4



3番マシンを停止、仕事は他マシンへ退避



3番マシンの停止を検出



3番マシンを再起動(3番の画面)



3番に仕事が移動し、再び定常状態に

自由市場原理に基づくスケジューリング方式

全知全能である必要があるグローバルスケジューリング方式を諦め、マシン間のスケジューリングはアプリケーションプログラムの自由意思に任せることにする

自己責任でスケジューリングの判断を下すための情報を低コストでアプリケーションプログラムが獲得できる必要がある

公平性は確保されねばならない、つまり他人に不合理な迷惑を掛ける自由は認めない

「厚かましい」アプリケーションプログラムが結果的に優遇されることを避ける必要がある

自由市場原理に基づくスケジューリング方式
Free Market Mechanism 方式 (FMM方式)

自由主義方式のスケジューリング

低コストでアクセスできる情報開示機構

公平性を守るための基本ルールの確立

ユーザプログラムのFMM用基本戦略

1. 適当な粒度（数百msec）で本来の仕事を実行
2. 情報開示機構に問い合わせて一番軽いマシンM1を調べる
3. M1と自マシンの負荷を比較し、移ることが得かどうか判断
4. 非移動が得である場合は1.に戻る
5. サイコロを振り、移送猶予期間 T1をランダムに決定
6. さらに T1の時間を過ぎるまでの間、本来の仕事を実行
7. 情報開示機構に問い合わせて一番軽いマシンM2を調べる
8. M2と自マシンの負荷を比較し、移ることが得かどうか判断
9. 非移動が得である場合は1.に戻る
10. M1とM2が一致の場合、移送システムコール発行、戻ったら1.へ
11. M1とM2が不一致の場合、M2を新たなM1として5.へ

並列アプリケーションの並行実行



SSS-POVRAY



SSS-POVRAY

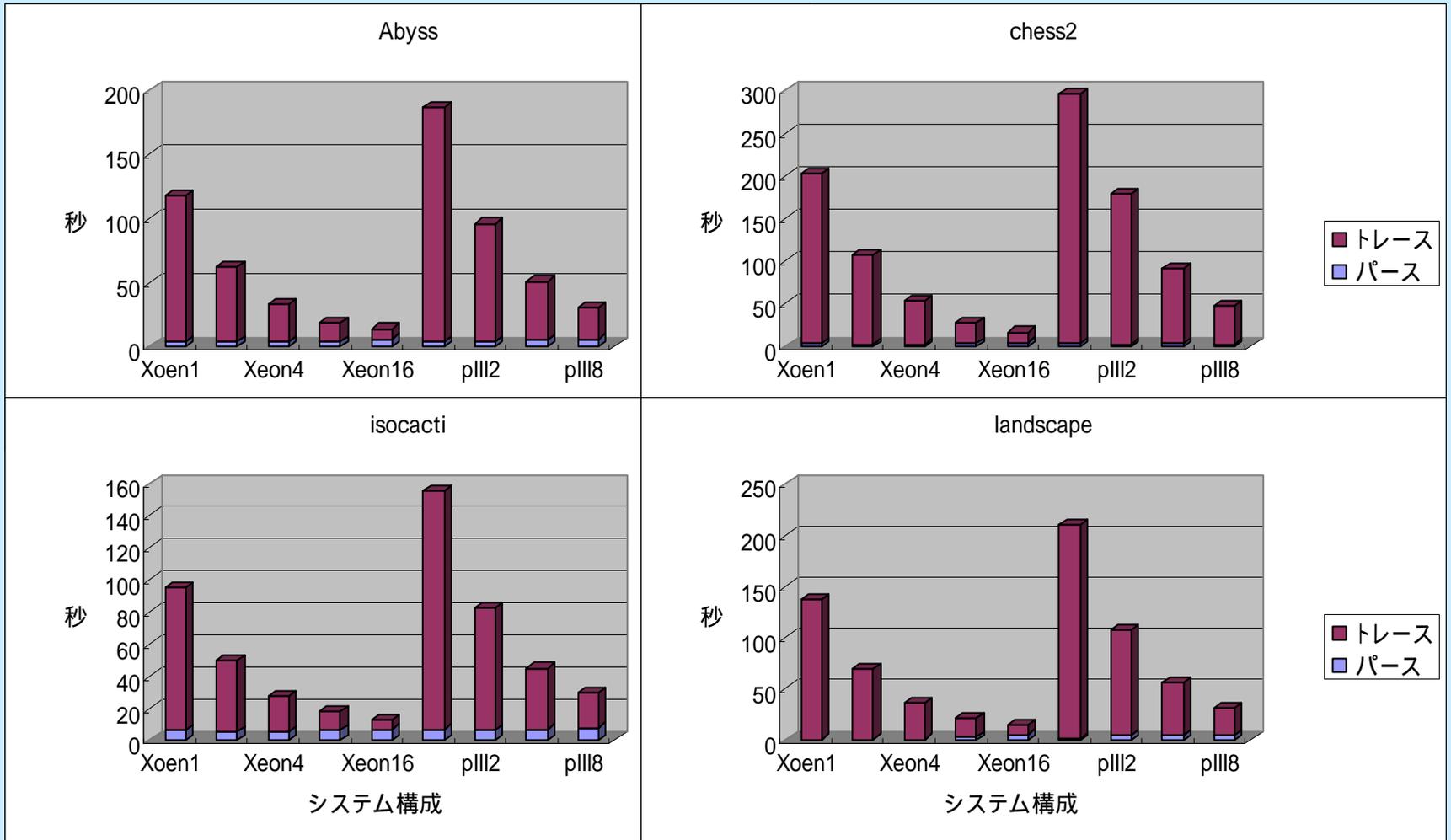
SSS-PC 1: 206

	SSS-POVRAY	SSS-POVRAY	libconv
1: lib	●	●	●
2: scen1	●	●	●
3: scen2	●	●	●
4: scen3	●	●	●
5: scen4	●	●	●
6: scen5	●	●	●
7: scen6	●	●	●
8: scen7	●	●	●
9: scen8	●	●	●
10: povray0	●	●	●
11: povray1	●	●	●
12: povray2	●	●	●
13: povray3	●	●	●
14: povray4	●	●	●
15: povray5	●	●	●
16: povray6	●	●	●
17: povray7	●	●	●

```

!! Task#77764:046011042 Finished !! (cmd="wcm",ret=0)
c:\ss-povray@node0 id=0 start.
wcm r_rt 0
arg=0 arg=[wcm r_rt 0]
c:\wcm r_rt(2406000) 0
set 2406000 %screen [0]
start 1 2 3 4 5 6 7 8 9 10 -- 17:povray pov -GA scenes/advanced/alpha.pov
arg=23 arg=[start 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 : povray pov -GA
scenes/advanced/alpha.pov]
set Mon3-Scrn No.
ss-povray@node1 id=1 start.
ss-povray@node2 id=2 start.
ss-povray@node3 id=3 start.
ss-povray@node4 id=4 start.
ss-povray@node5 id=5 start.
ss-povray@node6 id=6 start.
ss-povray@node7 id=7 start.
ss-povray@node8 id=8 start.
ss-povray@node9 id=9 start.
ss-povray@node10 id=10 start.
ss-povray@node11 id=11 start.
ss-povray@node12 id=12 start.
ss-povray@node13 id=13 start.
ss-povray@node14 id=14 start.
ss-povray@node15 id=15 start.
!! Task#77422:0440111e2 Finished !! (cmd="wcm",ret=0)
c:\ss-povray@node16 id=16 start.
ss-povray@node17 id=17 start.
    
```

POVRAYによる台数効果



XEON 2.8GHz E7500、PIII 1.26GHz Server Set III HE-SL、640x512ピクセル

ディペンダブルコンピューティングの基盤技術へ

- 冗長多重実行による多数決実行
- 突発事故によるマシン停止でもアプリケーションの実行継続
- キラーアプリケーション(並列サーバーアプリ)の作成
- SSS-PCの完成度の向上



「マネキツ」