

OSC2005資料

日本OSS推進フォーラムと開発基盤WGの取り組み

OSS性能・信頼性評価、障害解析ツール開発 プロジェクト成果の概要

発表資料: <http://www.ipa.go.jp/software/open/forum/>

2005年3月25日

日本OSS推進フォーラム
開発基盤WG 主査
鈴木友峰

(株)日立製作所 ソフトウェア事業部
OSSテクノロジセンタ

本プロジェクトの成果は、独立行政法人 情報処理推進機構 (IPA)
オープンソースソフトウェア活用基盤整備事業に係る委託業務の一環として開発しました。



1. 日本OSS推進フォーラムの概要 (1) 設立背景と目的



2004年2月設立。幹事団7名、顧問団13名(企業・団体のトップ、学識経験者)で構成

設立背景

1. 2003年9月の日中韓IT担当大臣合意、
11月の日中韓オープンソースビジネス懇談会の成果
 - 日本を代表してOSSについて国際協力していく団体が必要
 - OSS普及について民間の意見を集約する場が必要
2. OSSのシステムへの適用の進展
 - ユーザが安心してOSSを利用するための技術的、制度的課題解決の必要性

設立目的

1. 政府、民間で協力することによる日本国内でのOSS普及拡大
2. ユーザが安心して使えるための技術的、制度的課題の解決と新たな選択肢の提供
3. 日中韓、世界のコミュニティとの協調によるOSS発展への貢献

1. 日本OSS推進フォーラムの概要 (2)組織



オブザーバ

経済産業省
総務省
JISA *1

事務局

IPA *2

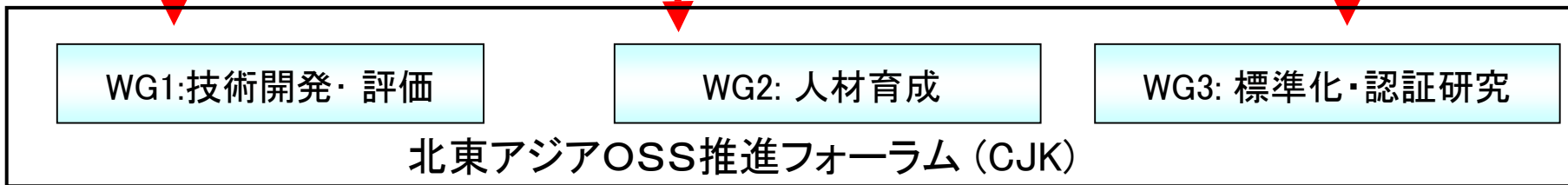
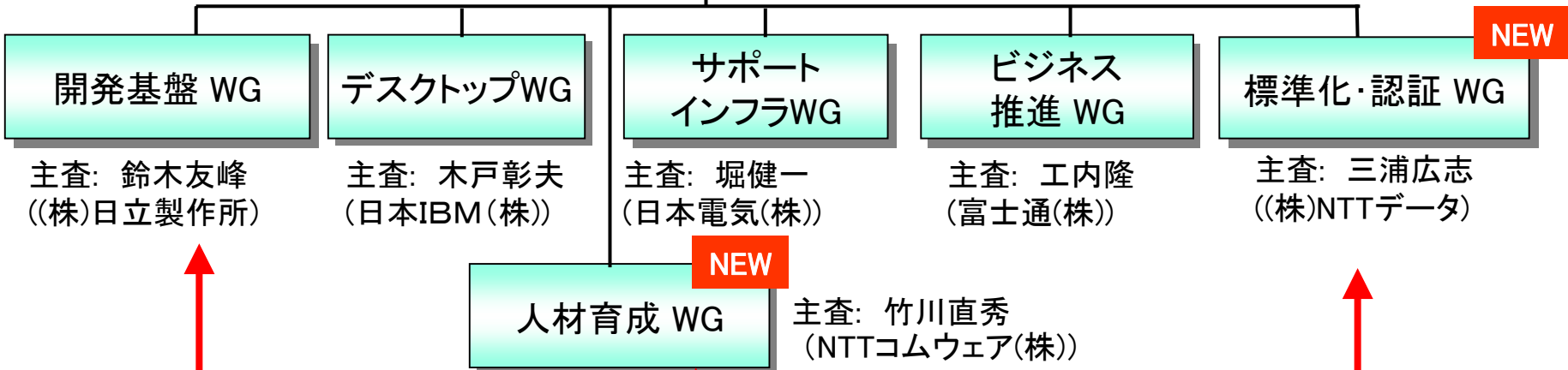
SC小委員会

幹事団 / 顧問団
代表幹事: 桑原洋
((株)日立製作所 取締役、前総合科学技術会議 議員)

ステアリングコミッティ(SC)
座長: 山田伸一
((株)NTTデータ 取締役)

*1: 社団法人情報サービス産業協会
*2: 独立行政法人 情報処理推進機構

CE Linux Forum
組込 Linux
協調



1. 第3回北東アジアOSS推進フォーラム(12/2-3ソウル)の成果



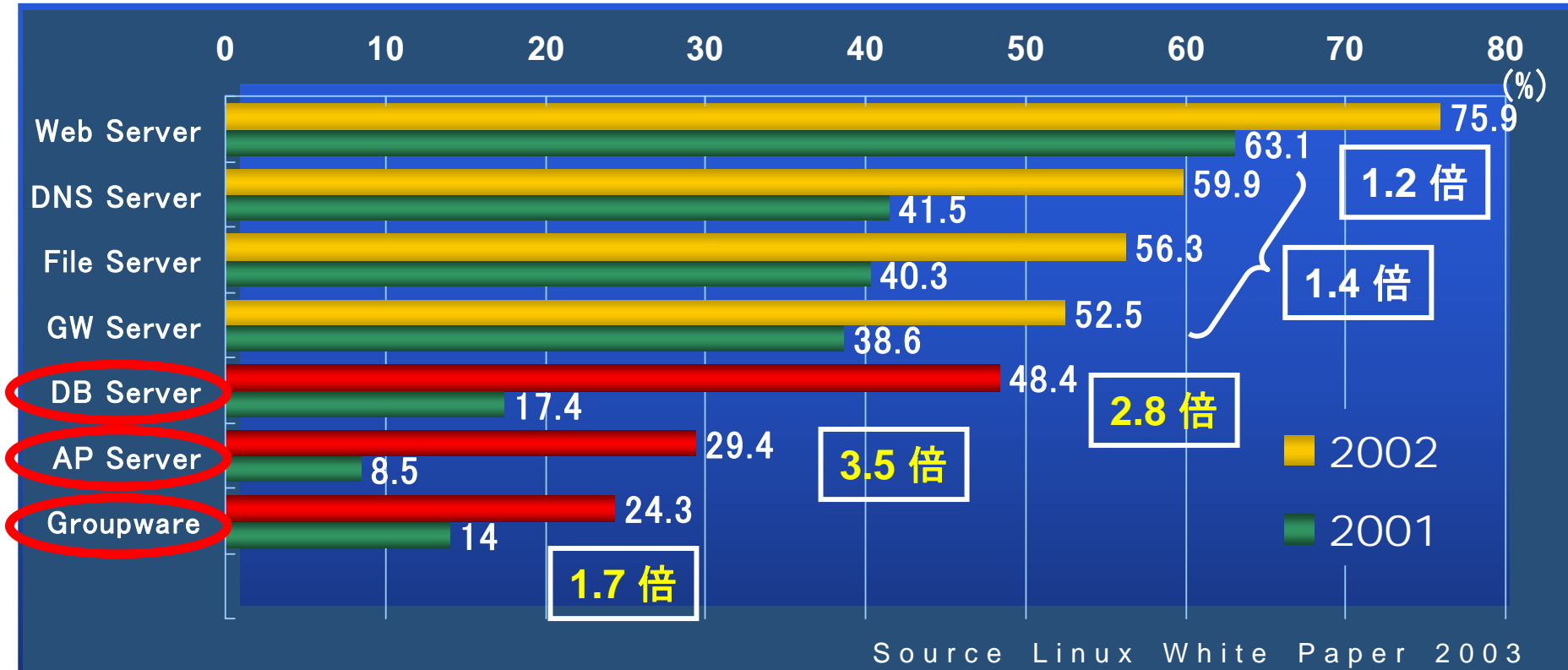
第3回 北東アジアOSS推進フォーラムでの合同WGに関する合意事項(議長声明内容)	
WG1	<p>1) 次の技術分野において協力する。</p> <ul style="list-style-type: none">デスクトップLinuxLinuxとその他のOSSのベンチマーク評価オペレーティングシステムのセキュリティ <p>2) 次の3分野において評価を実施し、結果を共有する。</p> <ul style="list-style-type: none">中国 - デスクトップLinux日本 - Linuxとその他のOSSのベンチマーク評価韓国 - オペレーティングシステムのセキュリティ <p>3) 評価の結果に基づき、協力開発の対象と方法の議論を継続する。</p> <p>4) 進捗確認のために3ヶ月毎に情報交換を実施する。</p>
WG2	<p>1) 調査とコンテストのための2つのタスクフォース(TF)を設置する。</p> <p>2) TF1は、OSSの教育と研修に関する目的、手法、範囲、成果共有の方法等の調査の枠組みを定める。</p> <p>また、本TFは「貢献者」とその計測手法を定義し、貢献者の増加のための手段を講ずる。</p> <p>3) TF2は2005年北京フォーラムへの準備としてOSSコンテストの計画を開始する。</p> <p>4) WG2憲章を制定した。</p>
WG3	<p>1) WG3憲章を制定した。</p> <p>2) WG3の指示書(directive)を作成するドラフティングコミッティを設置することを決定した。</p> <p>3) 次の領域について短期活動項目を定義した。</p> <ul style="list-style-type: none">インプットメソッド、標準化のための人材の育成、組込みLinux、Webデータのインターオペラビリティ



2. 日本のサーバLinux市場動向(1)



■ 企業内でのLinuxの用途は？



従来から使われてきたWebサーバ等のインターネット系サーバ用途から、DBサーバ、APサーバ等への適用が増加している

3. サーバ向けOSSの現状における課題



ベンダ、Slerから見た課題

ニーズがLinuxだけでなく、ミドルにまで拡大し、OSS適用システムが複雑化

それにもかかわらず...



- ・信頼性・性能等のシステム設計・構築に必要なデータが不足
(結果として、各社が同じような評価を実施)

- ・障害解析ツールが不足しており、原因究明に時間がかかる



4. プロジェクトの目的



プロジェクトの目的

サーバLinux、OSSの更なる普及・拡大のためのベンダサイドの課題解決
⇒各企業内にあるOSSノウハウのDB化とオープン化

1. **ベンダ共同のOSSの性能・信頼性評価によるシステム設計・構築ノウハウの共有**
 - －結果だけでなく、手順やデータも共有し、標準化を図る
 - －広くコミュニティに公開することで、OSSの普及に貢献
 - －ベンダにおけるOSS評価コストの低減(特にカーネル2.6、AP層、DB層などの新分野)
 - －多様なノウハウをベースとしたシステム構築によるシステム信頼性向上
2. **障害情報解析ツールの開発(例えばダンプ解析、トレーサ等)とノウハウの共有**
 - －ツールの利用ノウハウのベンダ間での共有とブラッシュアップ
 - －必要な機能はプロジェクトで開発し、公開することでコミュニティに貢献
 - －障害解析時間の短縮
 - －ミッションクリティカルシステムへの適用ニーズに対応

5. プロジェクトのロードマップ



国内におけるサーバLinux、OSS普及のための「企業コミュニティ」の形成・育成・発展を目指し、以下のロードマップで推進

2004年度

フェーズ1 コミュニティの形成

1. 国内ベンダ、Sierの結集 & 民間技術者の連携意識の醸成
具体的実施事項：
 - ・共同ベンチマーク実施とノウハウの共有
 - ・高信頼化ツールの開発とノウハウの共有
2. 中韓連携の土壌開拓
具体的実施事項：
 - ・人脈の確立
 - ・共通意識の確立

2005年度

フェーズ2 コミュニティの育成

1. 高信頼化のための共同評価範囲拡大
追加実施事項：
 - ・評価指針(手順)の標準化
 - ・評価範囲の拡大(クラスタ等)
 - ・ベンチマークツール設計・開発
 - ・チューニングノウハウの共有
 - ・障害予防ツールの整備
2. 新しい人材の投入
 - ・参加企業拡大
 - ・学との連携
3. 中韓との共同開発検討

2006年度

フェーズ3 コミュニティの発展

1. 国内ベンダ共同評価成果の公開範囲拡大、利用ユーザの拡大
2. 適用ユーザ事例の公開
3. 中韓との共同開発

6. メンバと状況



「OSS技術開発コンソーシアム」をWGメンバ企業8社で形成し、
2004/10～2005/2まで、IPAの「OSS活用基盤整備事業」として委託を受け
具体的作業を実施

■ WG メンバ企業

-主査

-(株)日立製作所

-メンバ企業(コンソーシアムメンバ)

-(株)SRA.

-(株)NTTデータ

-新日鉄ソリューションズ(株)

-住商情報システム(株)

-(株)野村総合研究所

-ミラクル・リナックス(株)

-ユニアデックス(株)

-メンバ企業 (非コンソーシアムメンバ)

-NTTコムウェア(株)

-日本ユニシス(株)

-オブザーバ

-Novell, Inc., OSDL, Red Hat KK

-新メンバ(2005年1月から加入)

-富士通(株)

-日本電気(株)

-ターボリナックス(株)

-(株)日立システムアンドサービス

-日本HP(株)

-(株)テンアートニ

7. 実施事項の概要(1)



具体的な実施事項

1. ベンダ共同のOSS性能・信頼性評価による システム設計・構築ノウハウの共有

- ①Javaアプリケーション層の評価
- ②DB層の評価
- ③OS層の評価



- ・結果だけでなく、ツール・手順も共有
- ・広くコミュニティに公開

2. 障害解析ツールの開発と利用ノウハウの共有

- ①ダンプデータ解析ツール(Alicia)の開発
- ②カーネル性能評価ツール(LKST)の開発
- ③ディスク割当評価ツール(DAV)の開発



- ・障害解析時時間の短縮
- ・高信頼システム適用ニーズに対応

7. 実施事項の概要(2)



想定されるアウトプット(ベンチマーク評価)

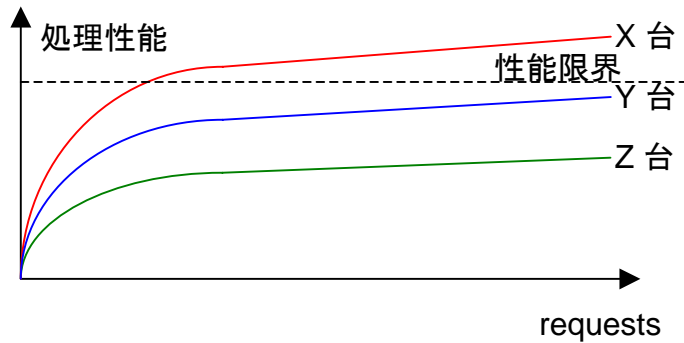
評価環境定義書

- ・評価HW、SW(OSS)構成
- ・評価ツール

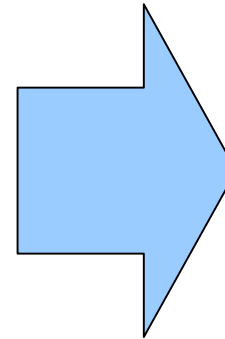
評価手順書

- ・SW(OSS)インストール、設定
- ・評価項目
- ・評価手順

評価結果



評価項目毎



公開、共有

OSS
Community



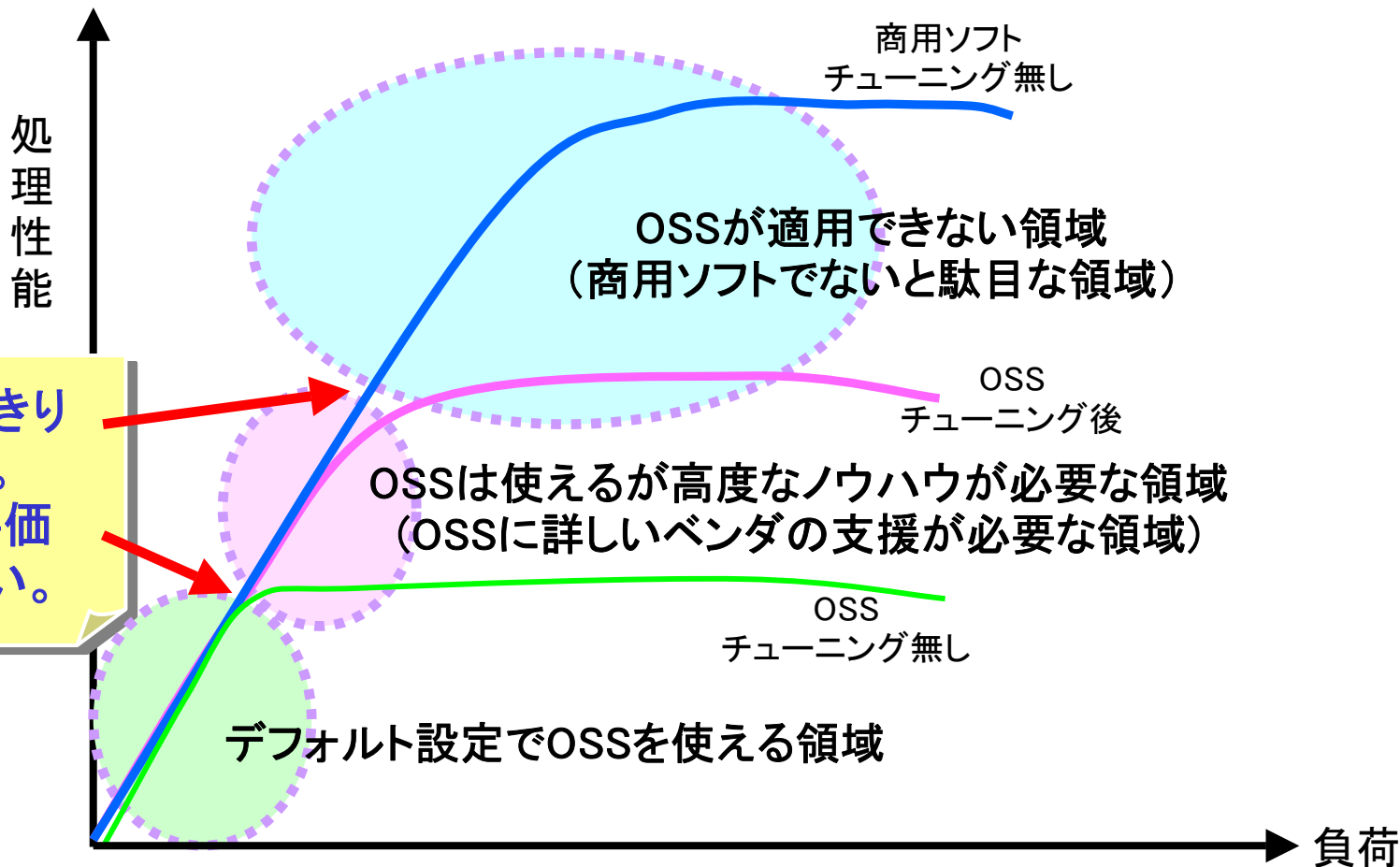
7. 実施事項の概要(3)



評価目的

OSSミドルが、現状でどこまで使えるかを、評価手順を明らかにしたうえで、明確化する
具体的にはOSSの処理性能はチューニングにより大きく変化するので以下3パターンを明らかにする

(1)デフォルトOSS、(2)チューニング後OSS、(3)商用



8. 成果の概要(1)



主な成果一覧(手順書関連)

1. Java AP層

- SPECjAppServer2004によるJBoss、WebLogic共通の性能評価手順を確立
(RedHatAS2.1,3、MIRACLE LINUX V3.0、SuSE Linux ES9+PostgreSQLでの手順を検証)

2. DB層

- OSDL DBT-1によるPostgreSQL、MySQL(MaxDB)、Oracle共通の性能評価手順を確立
(RedHatAS3、MIRACLE LINUX V3.0、SuSE Linux ES9での手順を検証)
- OSDL DBT-3によるPostgreSQL性能評価手順を確立
- 大量データのロードなど実SI場面を想定したPostgreSQL対応性能評価手順の確立

3. OS層

- CPU/IO負荷状態を想定したボトルネック解析手法の確立(iozone、oprofile、LKST)
- DBMSと相関を持つベンチマークの開発(diskio)

4. ツール

- ダンプ、トレーサ、ファイルシステムのフラグメンテーション可視化ツールのそれぞれについて、効果を測定し、障害解析手順をまとめた

8. 成果の概要(2)



評価手順をまとめたベンチマーク一覧

対象	ベンチマークツール	OS	対象ミドル	備考
AP	SPECj AppServer 2004	RedHatAS2.1、3 MIRACLE V3.0 SuSE9	JBoss4.0.0(RMS) WebLogic(MS)	・SPEC.orgが開発 ・有償(2K\$) ・EJB対応
DB	OSDL DBT-1	RedHatAS3 MIRACLE V3.0 SuSE9	PostgreSQL7.4(RMS) MaxDB(RMS) Oracle10g(M)	・OSDLが開発 ・TPC-W相当
	OSDL DBT-3	MIRACLE V3.0	PostgreSQL7.4、8.0	・OSDLが開発 ・TPC-H相当
	pgbench	MIRACLE V3.0	PostgreSQL7.4、8.0	・PostgreSQL付属 ・TPC-B相当
OS	Iozone	MIRACLE V3.0	—	・Iozone.orgから公開
	diskio	MIRACLE V3.0	—	・新規開発(DBサイジング用)

括弧内：R:RedHat、M:Miracle、S:SuSE

環境定義書・評価手順書の例



1.1 環境定義

1.1.1 システム構成

今回の評価では、PostgreSQL7.4.6用と8.0.0beta5用の、2つのシステムを使用して検証を行った。7.4.6用のシステムを表1.1-1に、8.0.0beta5用のシステムを表1.1-2に示す。

表 1.1-1 PostgreSQL7.4.6用のシステム

製品名	HP ProLiant DL380 Generation 3		
プロセッサ	インテル Xeon プロセッサ 2.80GHz、2CPU		
メモリ	2.5GByte		
ハードディスク	Ultra SCSI 320 HDD 15000rpm 内蔵ドライブベイに、3台		
ディスク構成	36GB HDD	/boot	1GB
		swap	4GB
		/	4GB
		/usr	4GB
		/var	1GB
		/tmp	1GB
		/home	20GB
	36GB HDD	/db_xlog	すべて
72GB HDD	/dbt3_0	すべて	

- ※ すべてのパーティションは ext3 ファイルシステムでフォーマットして、ext3 のジャーナリングのモードも、デフォルトの orderd モードを使用する。
- ※ 以下のディレクトリについては、その所有者を pgsq1 ユーザにしておく。
/db_xlog、/dbt3_0

表 1.1-2 PostgreSQL8.0.0beta5用のシステム

製品名	HP ProLiant DL380 Generation 3
プロセッサ	インテル Xeon プロセッサ 2.80GHz、2CPU
メモリ	2.5GByte

1.1.1.1 PostgreSQL のインストール

今回の評価では、バージョン 7.4.6 と 8.0.0beta5 を使用した。以下の手順でインストールできるが、二つのバージョンを同時にインストールする場合は、使用するディレクトリが競合しないように、適切に変更する必要がある。

root ユーザで、PostgreSQL の所有者となる Linux のユーザとして、pgsq1 ユーザを作成する。

```
# useradd pgsq1
```

PostgreSQL のソースコードアーカイブを展開するディレクトリと、PostgreSQL をインストールするディレクトリを作成して、ディレクトリの所有者を pgsq1 ユーザにする。

(a) PostgreSQL7.4.6 の場合

```
# mkdir /usr/local/src/postgresql-7.4.6
# chown pgsq1 /usr/local/src/postgresql-7.4.6
# mkdir /usr/local/pgsq1
# chown pgsq1 /usr/local/pgsq1
```

(b) PostgreSQL8.0.0beta5 の場合

```
# mkdir /usr/local/src/postgresql-8.0.0beta5
# chown pgsq1 /usr/local/src/postgresql-8.0.0beta5
# mkdir /usr/local/pgsq1
# chown pgsq1 /usr/local/pgsq1
```

pgsq1 ユーザで、PostgreSQL のソースコードアーカイブを展開し、展開したディレクトリに移動する。PostgreSQL のソースコードアーカイブは、/tmp ディレクトリにあるものとする。

(a) PostgreSQL7.4.6 の場合

```
# su - pgsq1
```

誰でも再現できる
レベルで記述

9. ベンチマーク評価結果の概要



評価結果からみた全体の考察(OSSの実システムへの適用可能性)

1. OSSは十分使える。ただし適用範囲を見極めて導入する必要がある

- ・JBoss4.0.0では、
 - ・比較的小規模なWebシステムに適用可能
 - ・クラスタ化することによってスケーラビリティも得られる
 - ・ただし、セッションリプリケーション機能等に不具合もあり、構築時に要注意
- ・PostgreSQL,MaxDBでは、
 - ・1つの目安として単体で秒間50～150トランザクション程度のシステムに適用可能
 - ・チューニングにより、性能は倍以上の向上が可能なので、個別システム毎に最適化して利用するのが有効

2. OSSは着実に進化している。

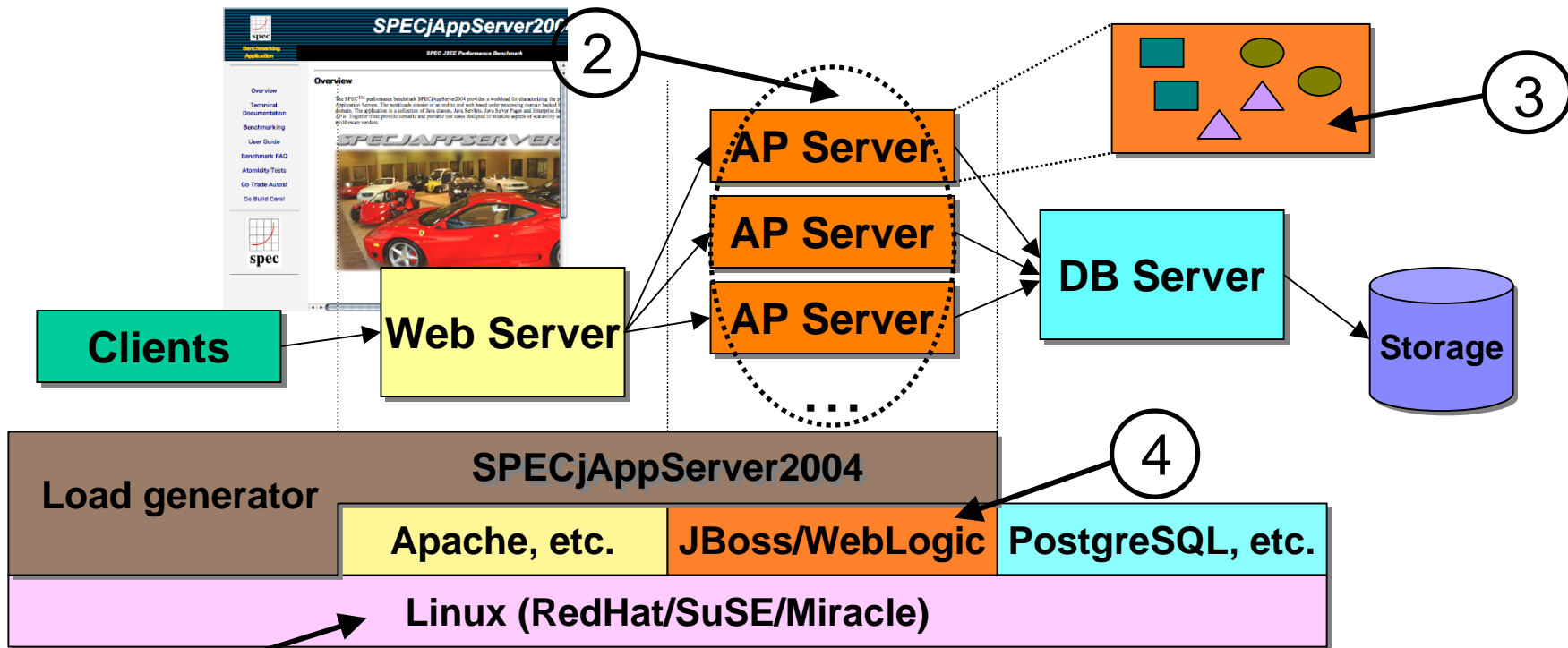
- ・Linuxカーネル
 - ・カーネル2.4に比べ、カーネル2.6では、高負荷時のレスポンス、安定性が向上している
- ・PostgreSQL
 - ・7.4と8.0では、10%程度の性能向上が見られる
 - ・PITR機能により、リカバリ時間が1/3程度に短縮されるのでこれを有効に活用すると良い

10. Java AP層評価結果 (1)評価項目



2004年度ベンチマーク評価項目

1. カーネル2.4と2.6の比較(2.6の性能や信頼性はどうか、等)
2. 分散処理性能の評価(大規模システムにおける性能・信頼性等)
3. トランザクション特性の解析(マイクロなレベルでの解析を行いたい)
4. 商用APサーバとの比較(JBossとWebLogicとの比較)



1

Javaアプリケーション層ベンチマークのSW/HW構成



2004年度のJava AP層評価の結果わかったこと(概要)

1. カーネル2.4と2.6の比較(単体測定)

- 単体性能比較において、高負荷時には2.6の方がやや高性能
- JBossは、高負荷時に急激に処理性能・応答時間等が低下
- JBossは、引数を値渡しにした際に、参照渡しに比べ性能が低い

2. 分散処理性能評価(クラスタ構成)

- JBossは、HTTPセッション複製を行う場合、性能が低い(バグ?)
- セッション複製をしなければ結構使える

3. トランザクション特性の解析

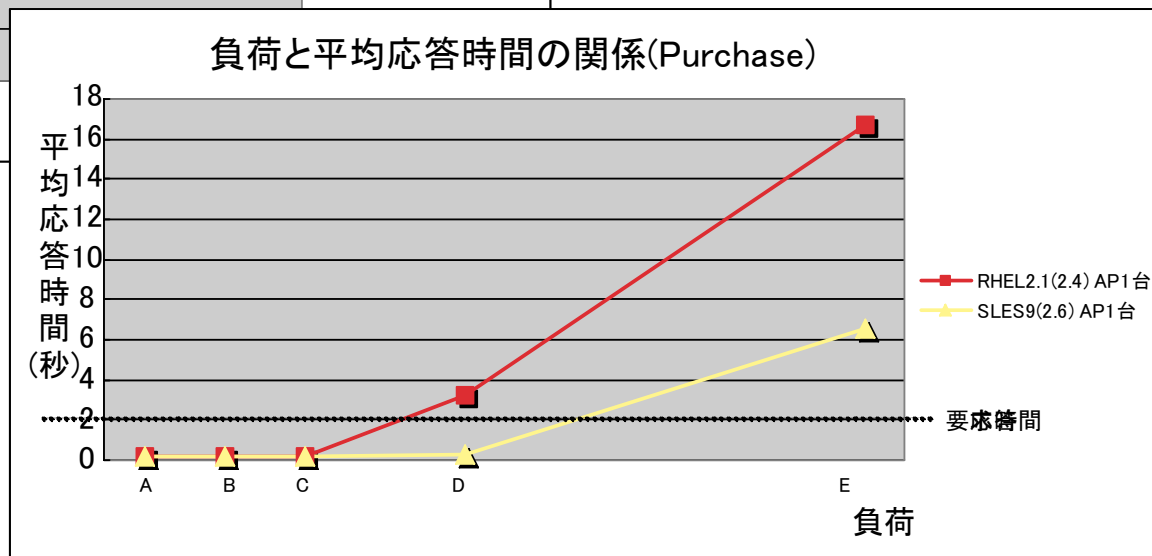
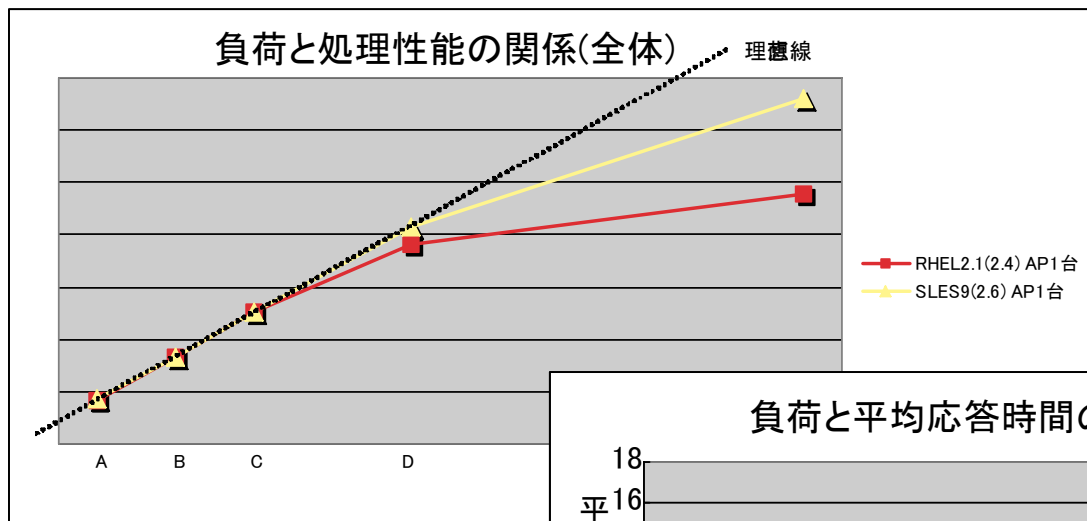
- ボトルネックになるメソッドが負荷・トランザクション種別により大きく異なる
(チューニングには、ボトルネックとなるメソッドの特定可能なツールが有効)

4. WebLogicとJBossの比較

- WebLogicはJBossに比べて高性能



カーネル2.4と2.6の比較

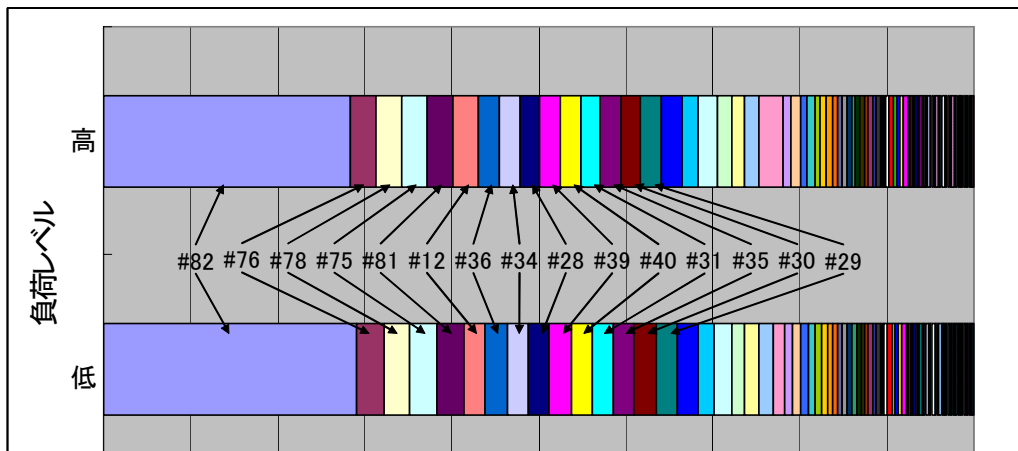


処理性能・平均応答時間共に、高負荷時には、カーネル2.6の方が、やや高性能
※:カーネル2.4: RedHatAS2.1、カーネル2.6: SuSE9

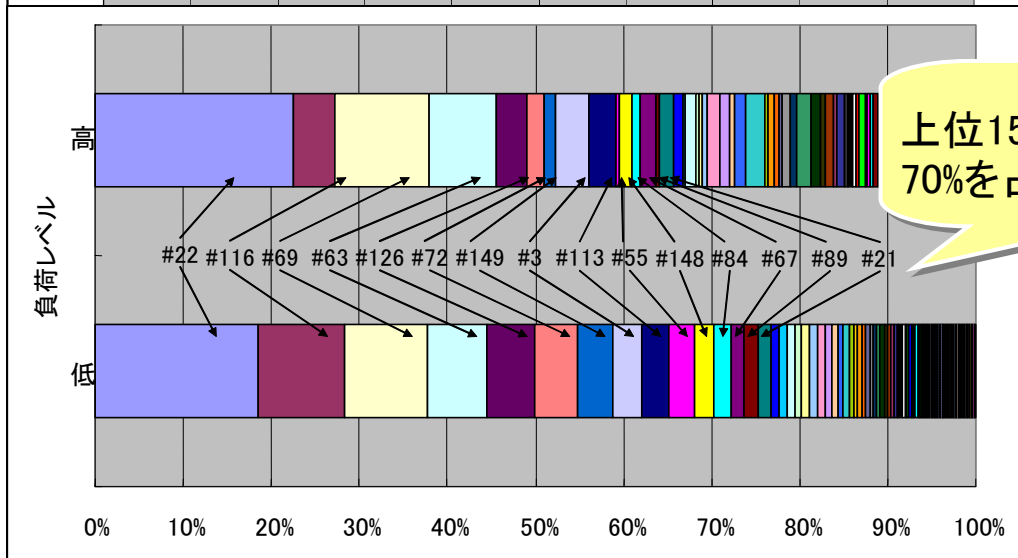


トランザクション特性の解析

メソッドレベルの
実行回数の割合



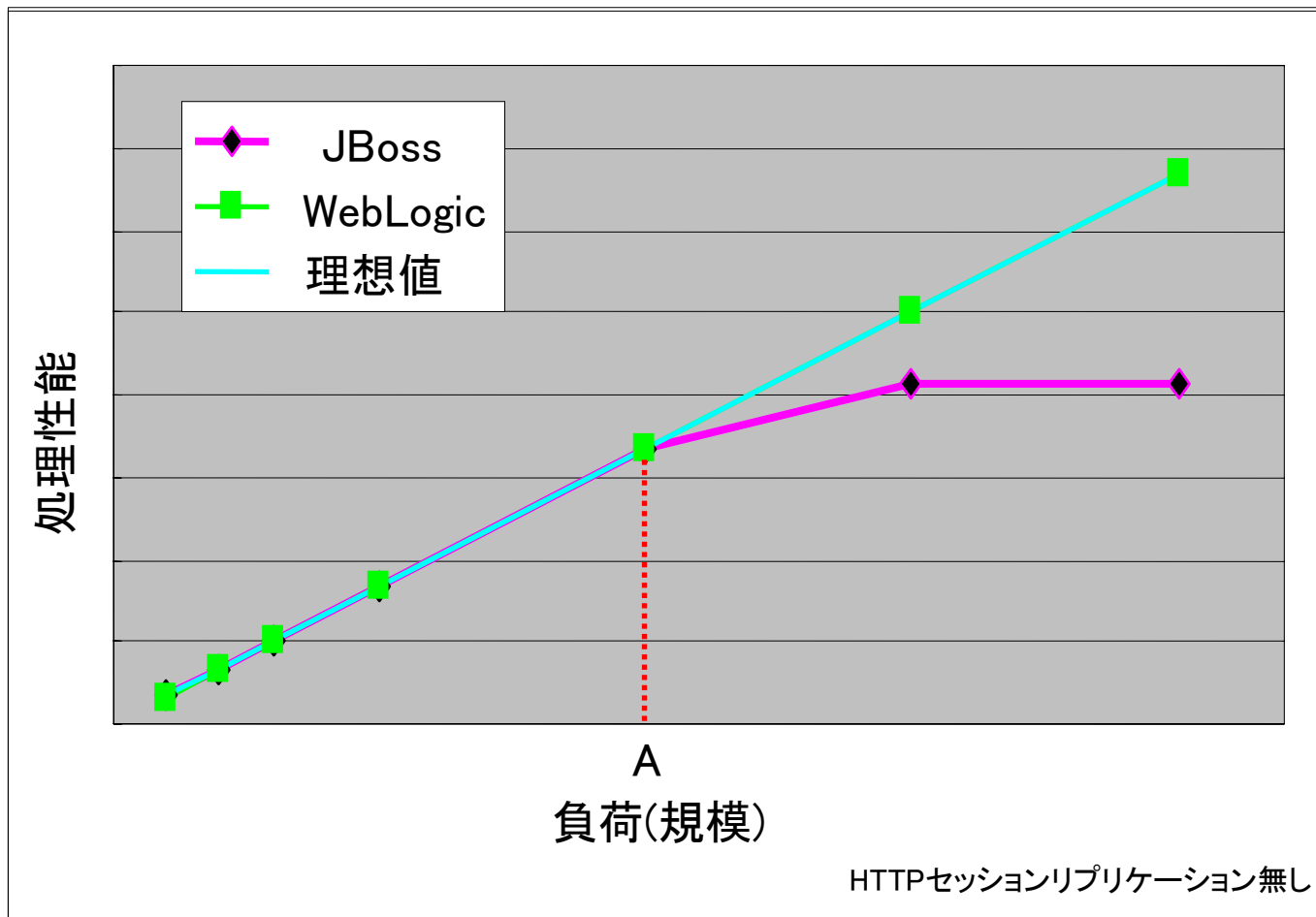
メソッドレベルの
実行時間の割合



メソッドの実行回数・時間の可視化により、ボトルネック解析と効率的なチューニングに有効



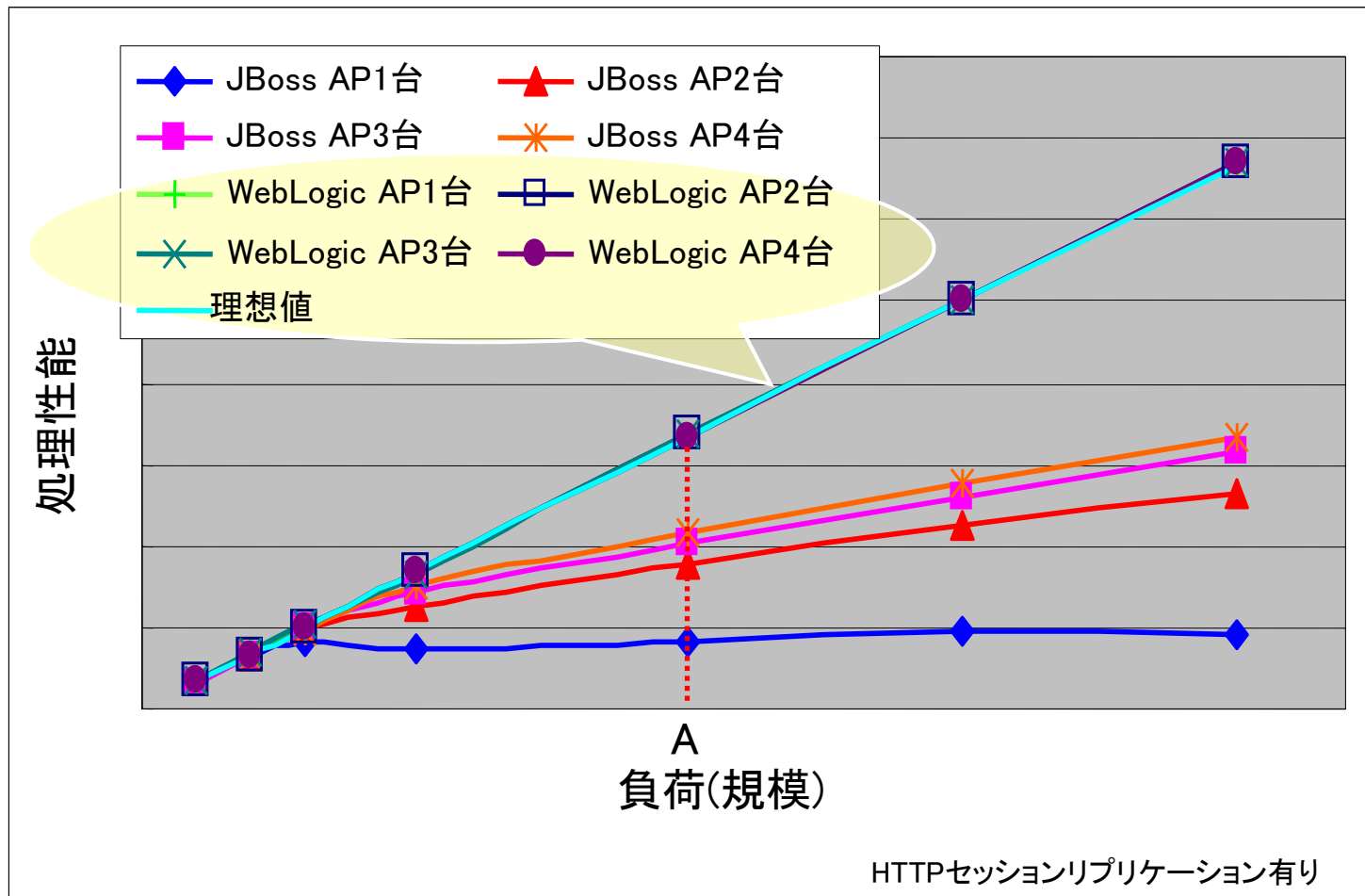
商用ソフトとの比較



測定した負荷の範囲内では、WebLogicの性能劣化は見られなかった(性能限界が高い)。JBossでは、負荷A程度までが限界で、それ以上の負荷ではレスポンスタイムが低下する。



商用ソフトとの比較



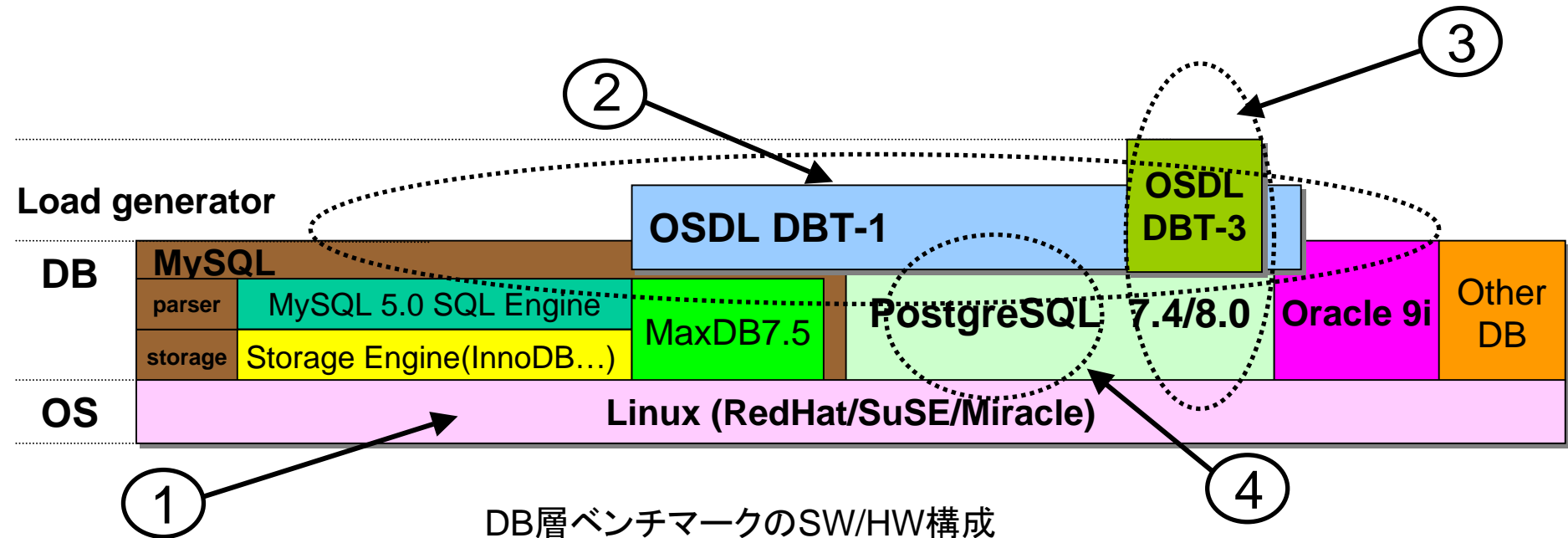
HTTPセッションリプリケーションありでは、JBossとWebLogicの性能差は拡大する。
JBoss4.0.2では改善される予定(バグ)。

11. DB層評価結果 (1)評価項目



2004年度ベンチマーク評価項目

1. カーネル2.4と2.6の比較(2.6の性能や信頼性はどうか、等)
2. Web系処理性能の評価(Webシステムにおける性能・信頼性、等)
3. DSS系処理性能の評価(DSSシステムにおける性能・信頼性、等)
4. 大規模DB性能(運用性)の評価(大規模データのバックアップ、ロード、インデックス再構成、等)



DB層ベンチマークのSW/HW構成



2004年度のDB評価の結果わかったこと(概要)

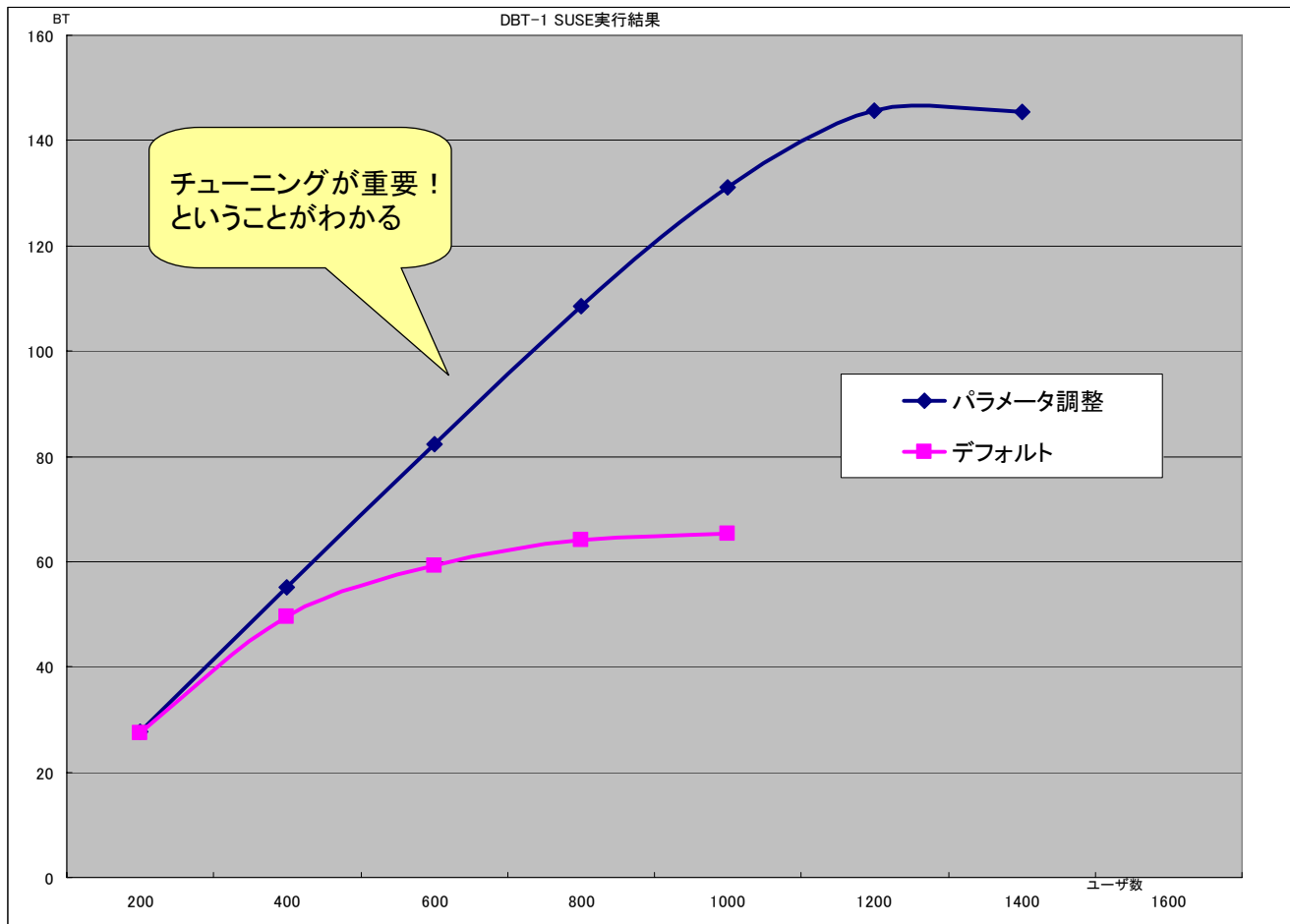
1. **カーネル2.4と2.6の比較・DBT-1による性能評価(Web処理系の評価)**
 - ・カーネル2.4に比べ、2.6では高負荷でも安定して動作する。
 - ・ディストリビューションによりCPU利用率、性能の特性が異なる
2. **DBT-3の性能評価**
 - ・DBサイズが4GBを越えた辺りから、性能への影響が一定となる傾向が確認できた
 - ・PostgreSQL 8.0では、ディスクを追加してI/Oを分散することによる問い合わせの性能向上の効果が確認できた
3. **大規模DB性能評価(PostgreSQL7.4と8.0の比較評価)**
 - ・大量データのロードは、ほぼデータ量に比例した時間がかかることが確認できた
(全体的にPostgreSQL8.0は7.4に比べ5～10%程度高速な傾向がある)
 - ・PITRによるリカバリはかなり高速であることが確認できた(回復時間が1/3になった)
4. **その他**
 - ・DBT-1の問題点が多発した(ファイルの欠如、コンパイルエラー、等)
 - ・チューニングの有効性が確認できた
 - ・ディストリビューションの差異が性能に影響することがわかった(スレッド実装、等)

11. DB層評価結果 (3)評価結果の詳細①



OSDL DBT-1の結果

-MaxDB7.5によるDBT-1のBT値(擬似トランザクション処理数)遷移



BT値

-simultaneous connection=1

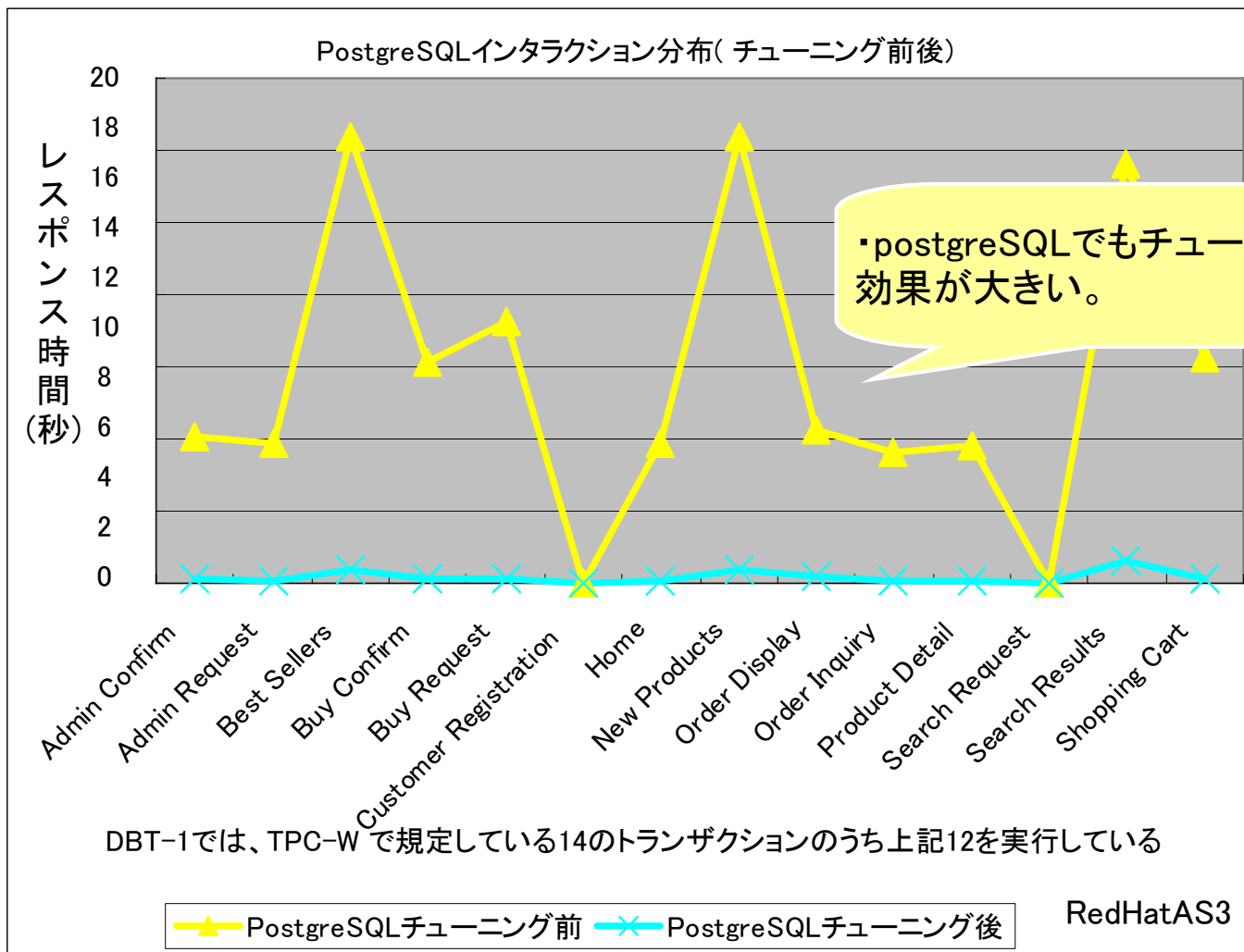
SUSE LINUX Enterprise Server 9
MaxDB7.5
CPU Intel Xeon 3.6GHz (HT) Dual
Memory 4GB

11. DB層評価結果 (3)評価結果の詳細②



OSDL DBT-1の結果

-PostgreSQLのチューニング前後の比較

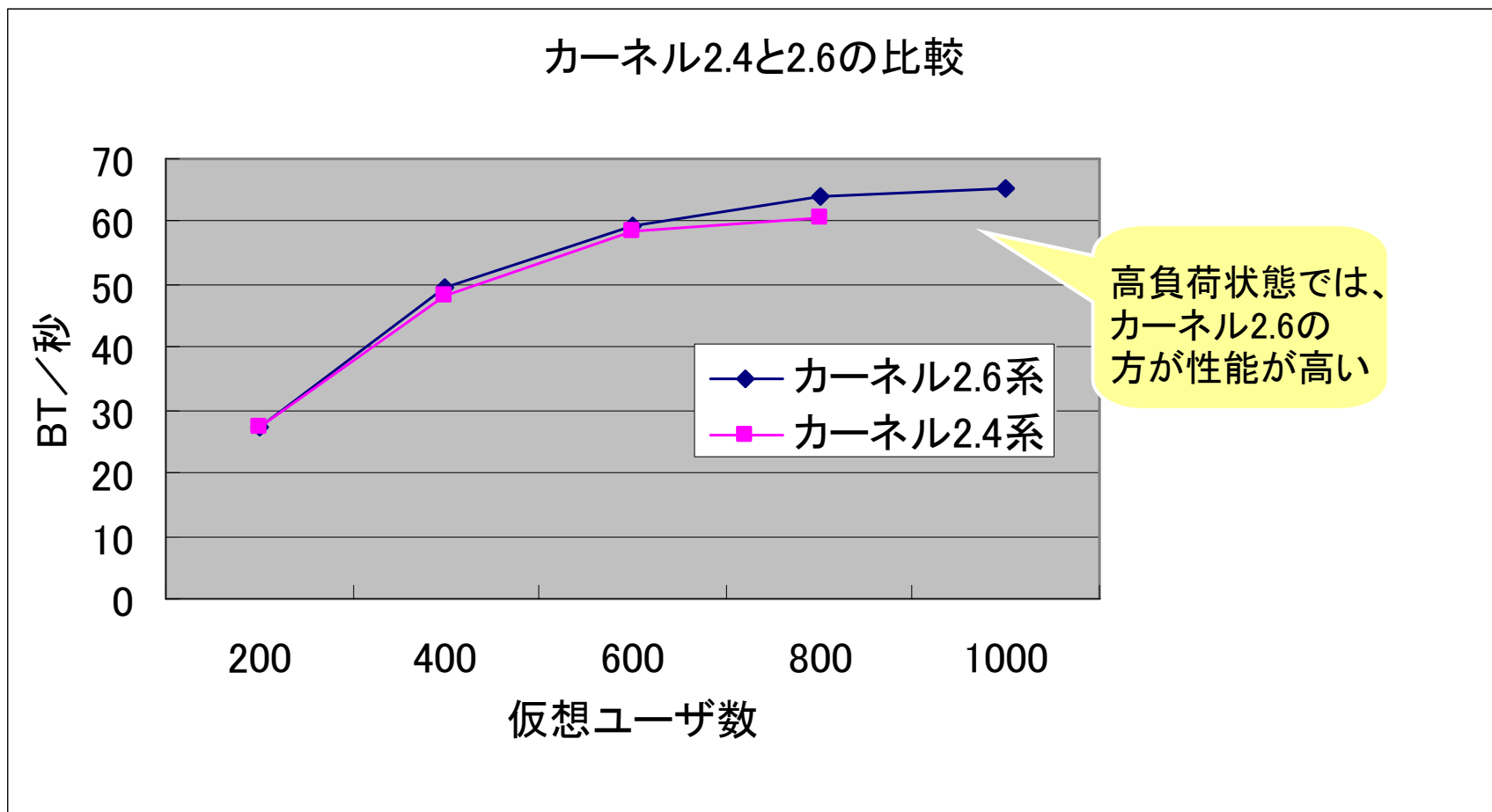


11. DB層評価結果 (3)評価結果の詳細③



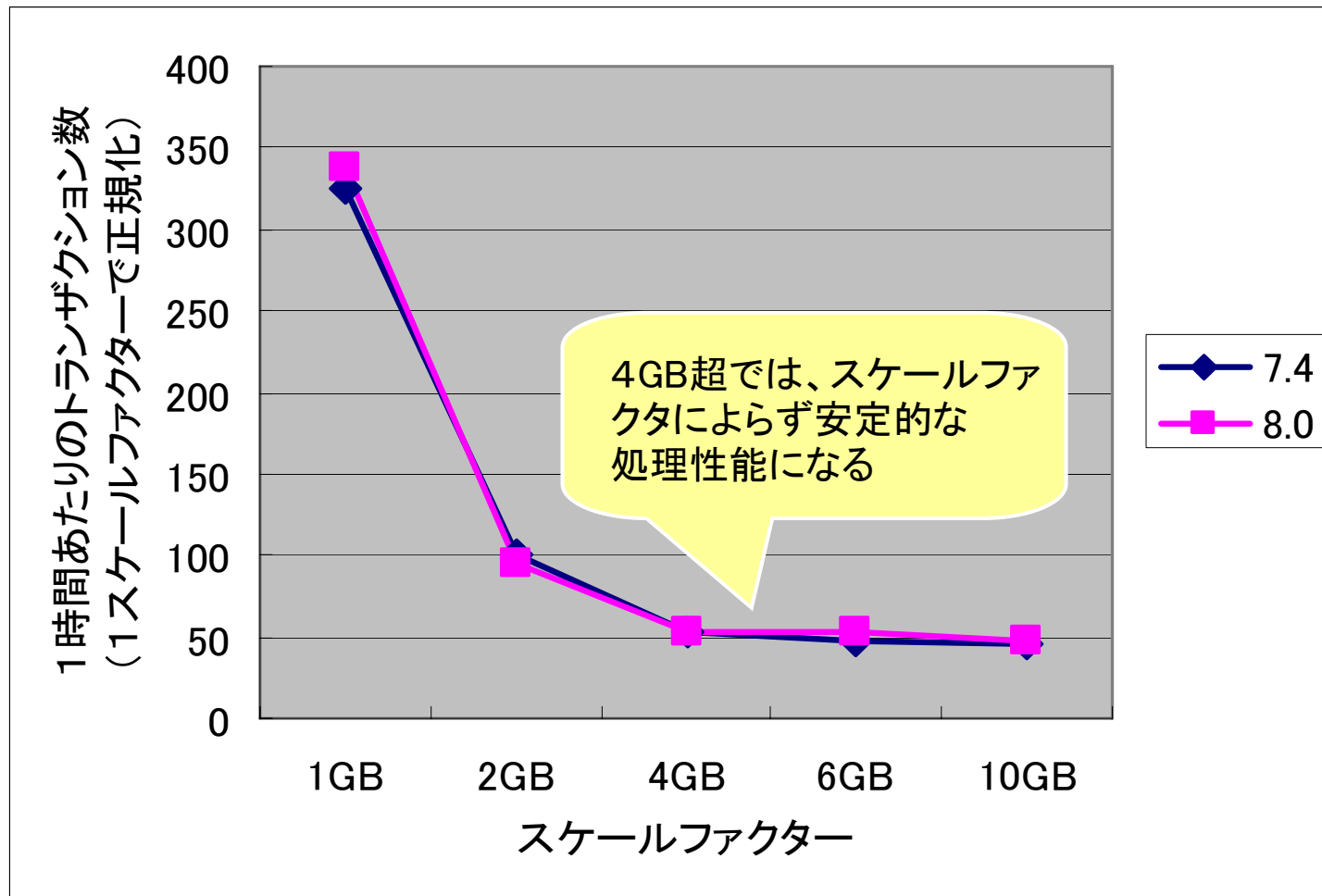
OSDL DBT-1の結果

—カーネル2.4と2.6の比較





OSDL DBT-3の結果

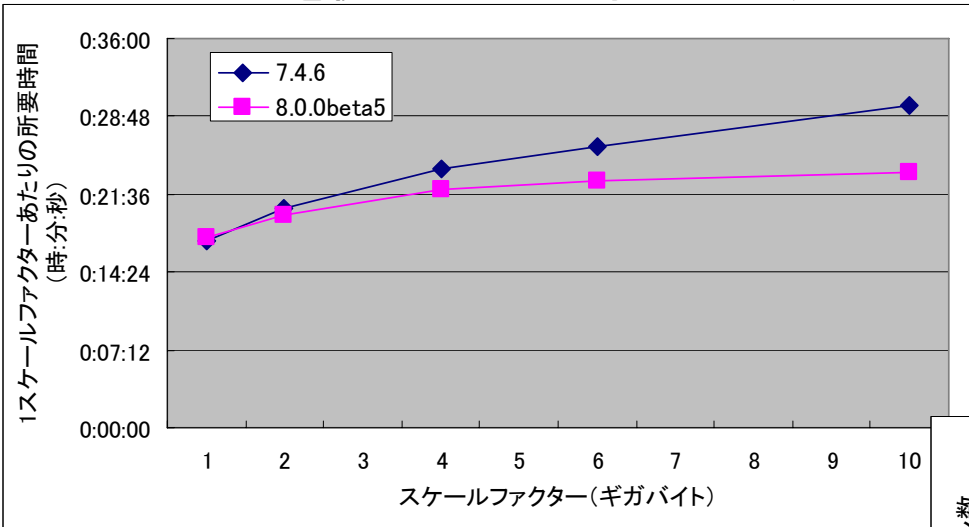


11. DB層評価結果 (3)評価結果の詳細⑤



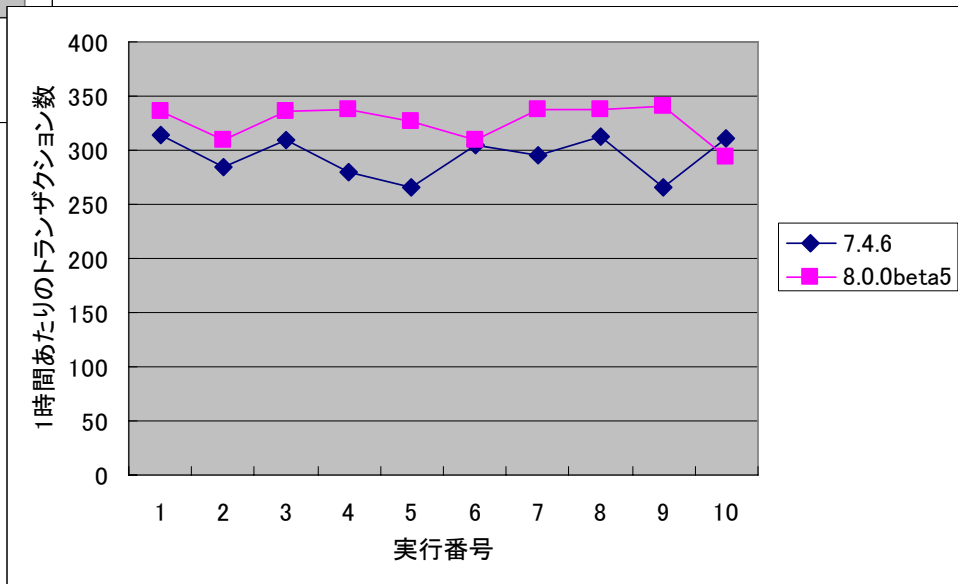
DBT-3によるPostgreSQL7.4と8.0の比較

DBT3を使ったロード時間の比較



PostgreSQL8.0では
着実な進歩が見られる

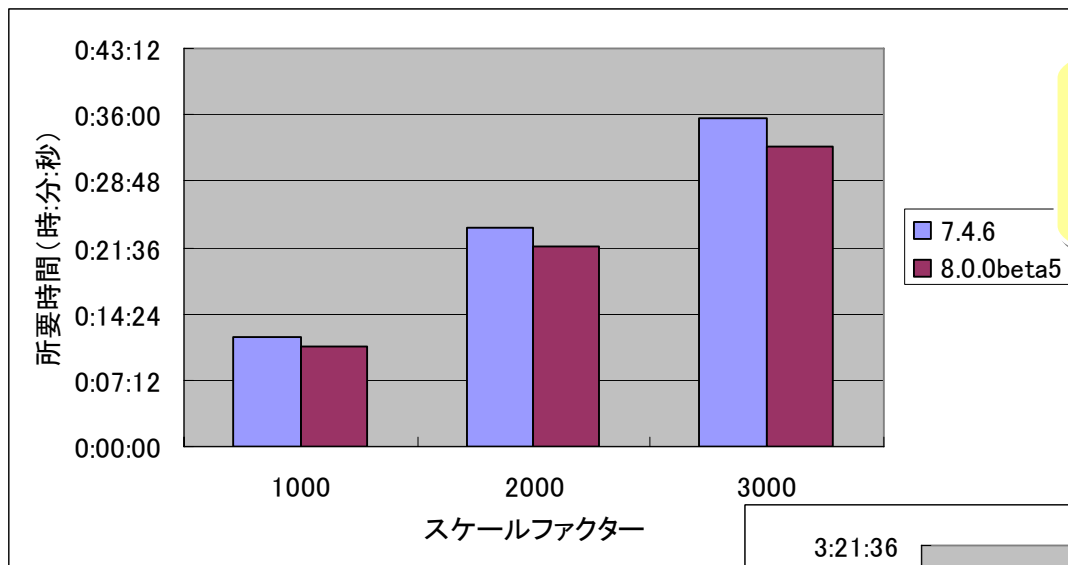
DBT3パワーテストによる比較



11. DB層評価結果 (3)評価結果の詳細⑥



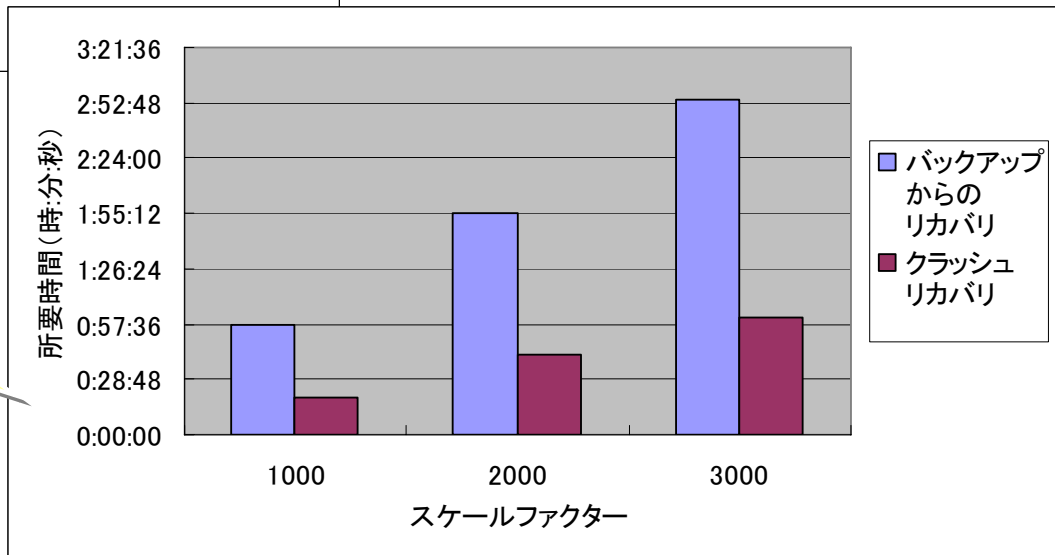
大規模データ時のPostgreSQL7.4と8.0の比較



ここでもPostgreSQL8.0では
着実な進歩が見られる

ークラッシュリカバリ時間比較
7.4 バックアップからの回復
8.0 PITRによる回復

PostgreSQL8.0でサポートされた
PITRにより、リカバリ時間は1/3に





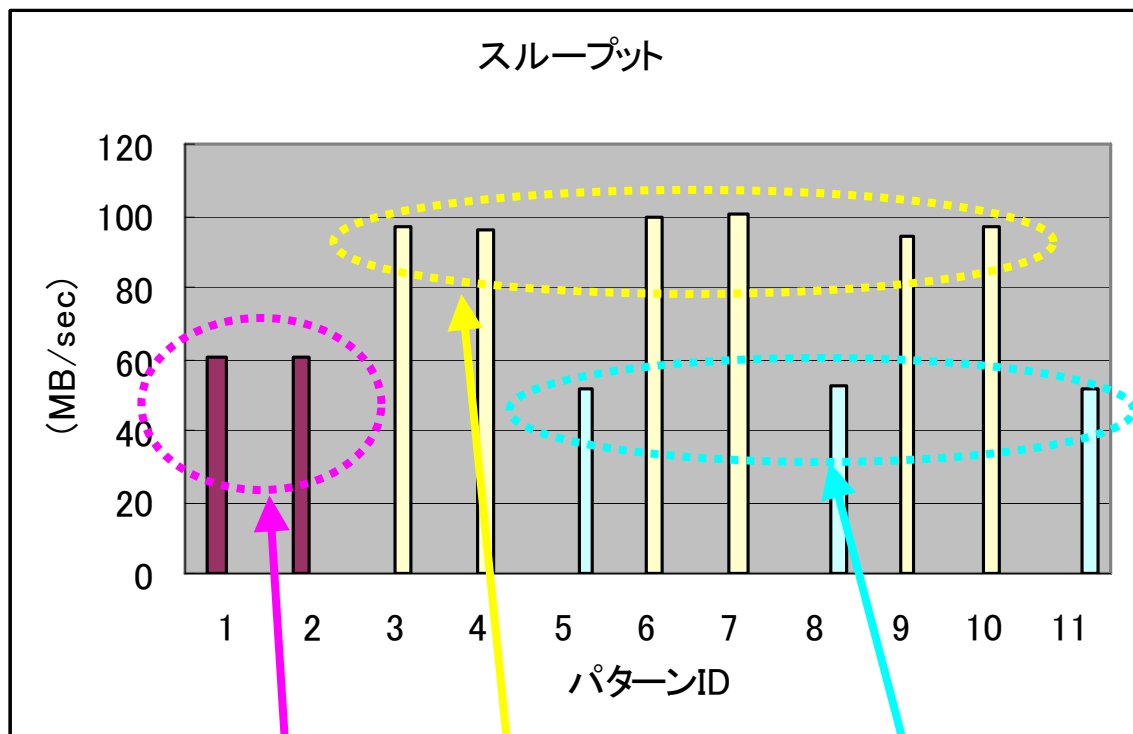
2004年度のOS評価の成果(概要)

1. Linuxカーネルの過負荷状態におけるボトルネックパターンとして、
①CPUネック、②I/O処理ネック、③ロック処理ネックの3パターンについて、
Oprofile、LKSTを使った解析手法を確立した
2. PostgreSQLのI/Oをモデル化したマイクロベンチマークを開発した(diskio)。
さらに、同ベンチマークとDBT-3の結果との相関を確認し、ディスクI/Oを
ミクロなレベルで解析する手法を提供した

12. DB層評価結果 (2)高負荷状態の解析①



Iozoneを様々なパラメータ(ID1~11)で実行し、高負荷状態におけるカーネル内部での挙動の違いをOprofile,LKSTを使って検証した



パターンID	ファイルサイズ	プロセスorスレッド数	スレッドの使用	iozoneコマンド実行数
1	8G	1	P	1
2	8G	1	T	1
3	4G	2	P	1
4	4G	2	T	1
5	4G	1	P	2
6	2.7G	3	P	1
7	2.7G	3	T	1
8	2.7G	1	P	3
9	2G	4	P	1
10	2G	4	T	1
11	2G	1	P	4

CPUアイドル型

CPUビジー型

ロック競合型

CPUアイドル型: 8GBのファイルのI/O待ちで発生(I/Oネック)

CPUビジー型: iozoneプロセスからページキャッシュへのデータコピー処理がネック

ロック競合型: グローバルなカーネルロックで発生(複数のiozoneコマンド起動時)

12. DB層評価結果 (2)高負荷状態の解析②



CPUアイドル型 (ID=1)とCPUビジー型 (ID=9)をOprofileで分析

1. CPUビジー型

Oprofileによるサンプリング結果(ID=9)

順位	シンボル	サンプル数	占有率(%)
1	do_generic_file_write	8,462	16.16
2	get_hash_table	4,584	8.754
3	unlock_buffer	3,364	6.424
4	__br_write_lock	2,763	5.276
5	__write_lock_failed	1,989	3.798

- ・上の結果から、do_generic_file_write() の実行頻度が一番高いことがわかる。
- ・細分化したところ0xc0160de8の命令が実行頻度が一番高いことが判明。
- ・ソースとの突合せの結果、_copy_from_user()が最も実行頻度が高いことが判明。
- ・_copy_from_user()はユーザ空間のデータをカーネル空間のページキャッシュに書き込む(コピーする)処理
⇒I/Oの中心処理なので、この処理の実行頻度が高いことは効率よい書き込み処理がされたことを示している

2. CPUアイドル型

Oprofileによるサンプリング結果(ID=1)

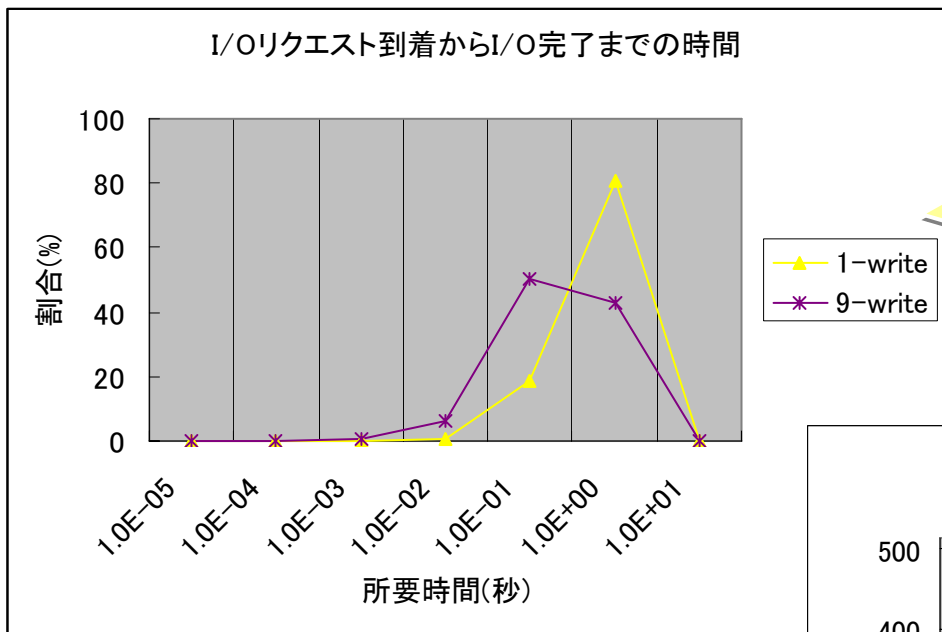
順位	シンボル	サンプル数	占有率(%)
1	default_idle	1,993	24.37
2	try_to_free_buffers	532	6.504
3	launder_page	532	6.504
4	LKST_ETYPE_MEM_SWAPOUT_HEADER_hook	371	4.536
5	unlock_page	365	4.462

- ・上の結果から、default_idle() の実行頻度が一番高いことがわかる。
- ・細分化したところhltの命令が実行頻度が一番高いことが判明。
⇒I/O待ちでCPUがアイドル状態であったことを示している

12. DB層評価結果 (2)高負荷状態の解析③

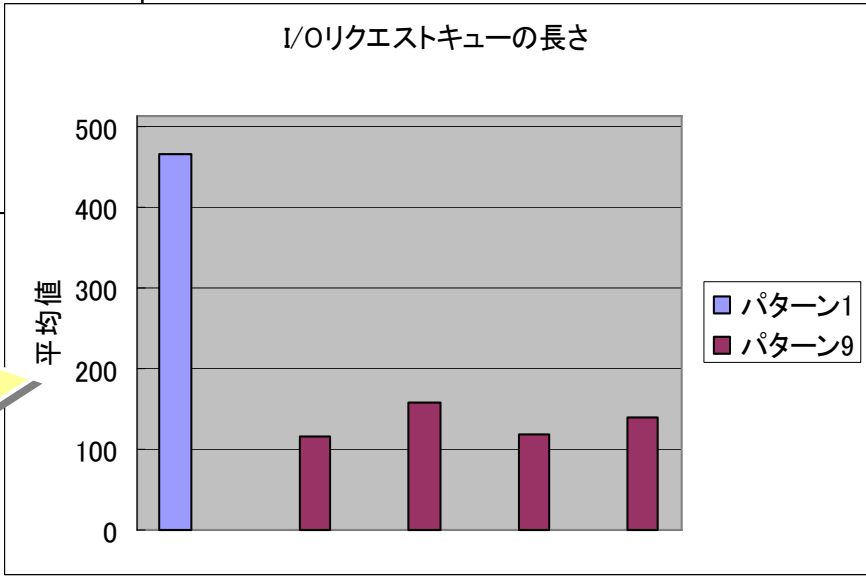


CPUアイドル型 (ID=1)とCPUビジー型 (ID=9)をLKSTで分析



CPUアイドル型(ID=1)では、CPUビジー型(ID=9)に比べ、I/O処理にかかる時間のピークが10倍違う(1秒vs0.1秒)

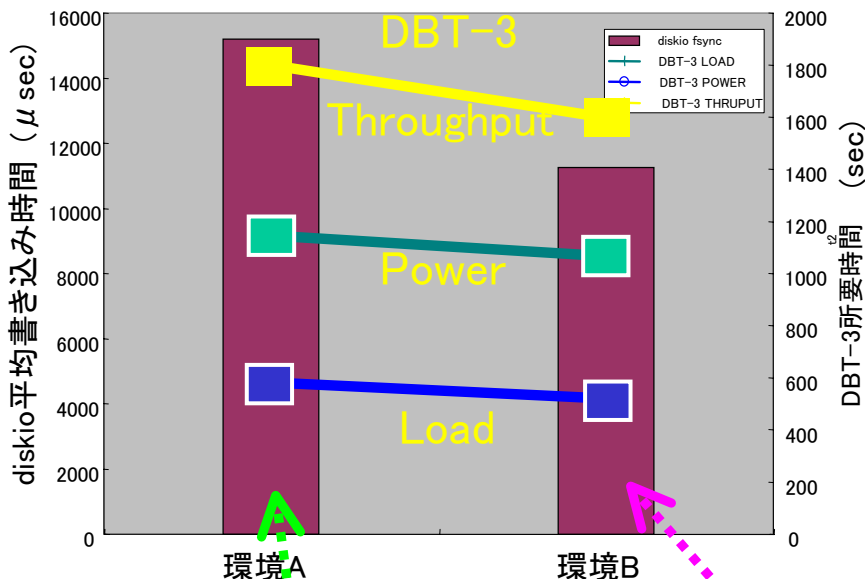
CPUアイドル型(ID=1)では、キュー長が最大の512に近いことから、ディスクの処理限界性能付近で動作していたことがわかる



12. DB層評価結果 (3)マイクロベンチマークの開発と評価



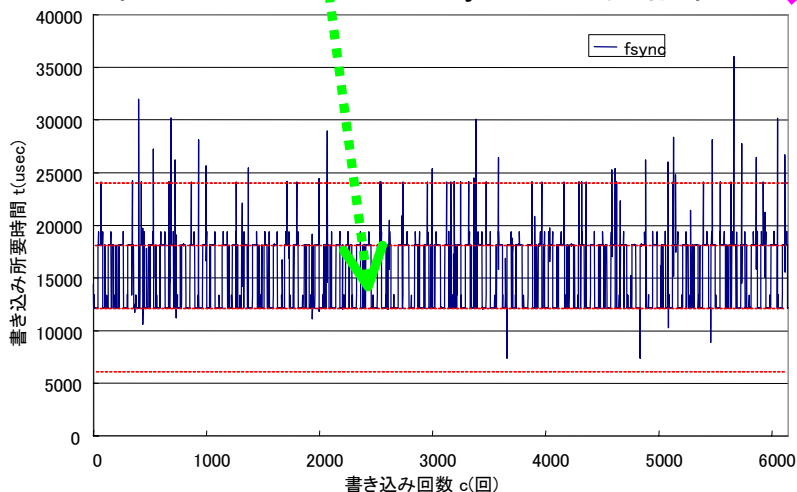
PostgreSQLのI/Oをモデル化したマイクロベンチマーク(diskio)の開発と評価



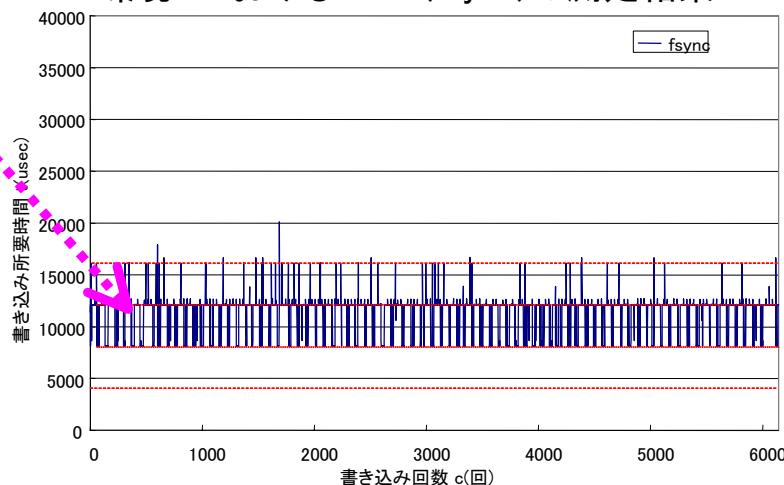
- 環境AのHDD : 10,000rpm
⇒1回転にかかる時間 = $60/10,000 = 6,000 \mu sec$
- 環境BのHDD : 15,000rpm
⇒1回転にかかる時間 = $60/15,000 = 4,000 \mu sec$
- Intel Xeon 2.8GHz × 2、メモリ 4GB(環境A),2.5GB(環境B))
- 下図から同期I/O(fsyc)ではほぼ2または3回転で書き込み完了している。これに対し、非同期I/O(async)では、fsyncに比べて3桁高速であった。

⇒fsyncでDBT-3実行時間を予測し
左図の通り相関を確認した

環境Aにおけるdiskio(fsyc)の測定結果



環境Bにおけるdiskio(fsyc)の測定結果





既存解析ツールに対する統合型ダンプ解析ツールAliciaの開発

ノウハウ蓄積:

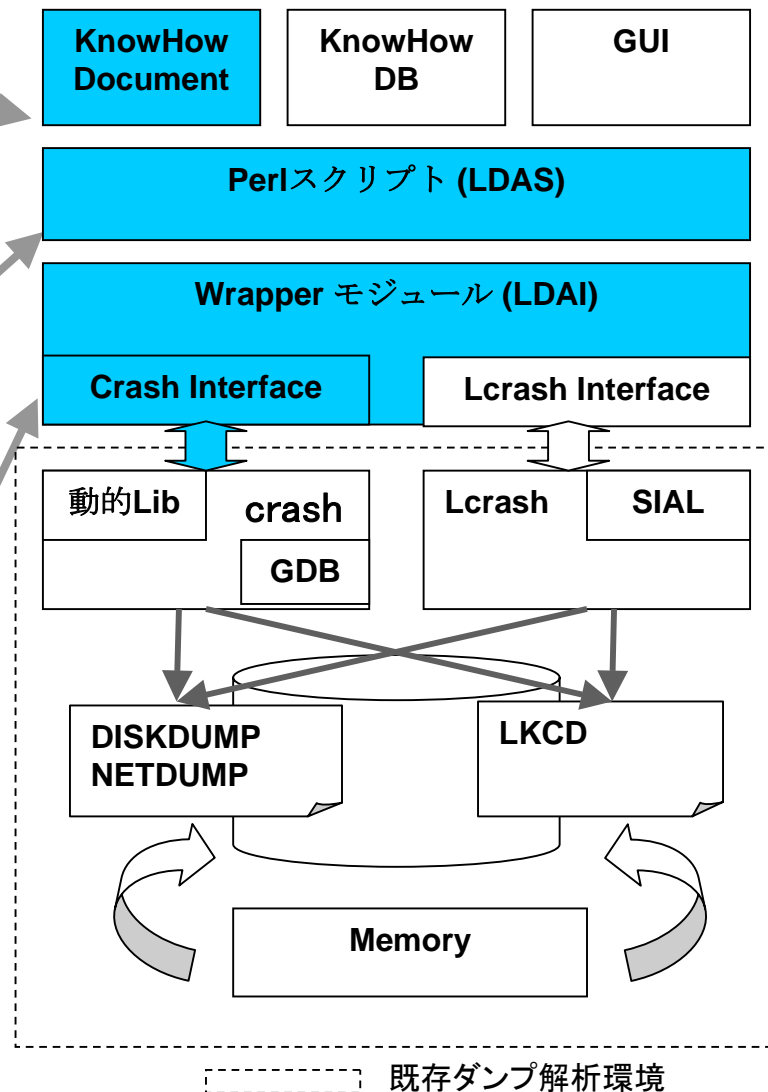
- LDASはインタプリタであり、可視性が高く、そのままDump解析のノウハウとして蓄積可能
- 修正も容易であり、Dump解析中に既においたスクリプトを変更して、新しい解析コマンドを即座に作成可能

Perlスクリプト(LDAS) (Linux Dump Analysis Scripts)

- Wrapperモジュールを利用して新規解析コマンドを開発
- これまでにDump解析で必要とされた手順をスクリプト化することで同種のトラブルを迅速に解析

Wrapperモジュール(LDAI) (Linux Dump Analysis Interface)

- 既存Dump解析ツール(Crash、Lcrash)コマンド群に対する入出力インターフェースの開発
- 既存Dump解析ツールが持っている機能を利用して、Kernel構造体へのアクセスを実現 (Alicia API)





2004年度の開発結果の評価

- **ダンプ解析ノウハウを蓄積/共有するための統合型ダンプ解析ツールとして実現**
 - Alicia本体部分とcrashインタフェースの開発を完了
 - 解析ノウハウを蓄積/共有するためのインフラを提供
 - いくつかの解析手順をスクリプト化し、蓄積/共有例として提示
- **Aliciaによるダンプ解析高速化の確認**
 - サンプル・スクリプトなしでの実行時間の測定
 - サンプル・スクリプト作成時間＋スクリプト実行時間の測定
 - サンプル・スクリプト再利用時の実行時間の測定
 - 操作性/利便性向上によるツール全体としての高速化の確認
 - リンクリスト・サーチ、カーネル停止直前のログメッセージの表示、CPU使用状況表示等のサンプル・スクリプトを作成して確認
- **企業間協力による開発成果 – シナジー効果を発揮**
 - 3社による協力体制により、開発速度、品質面のブラッシュアップに大きな効果を発揮
 - Alicia本体機能の開発範囲拡大と操作性/信頼性の向上を実現



Aliciaによる解析手順の高速化例

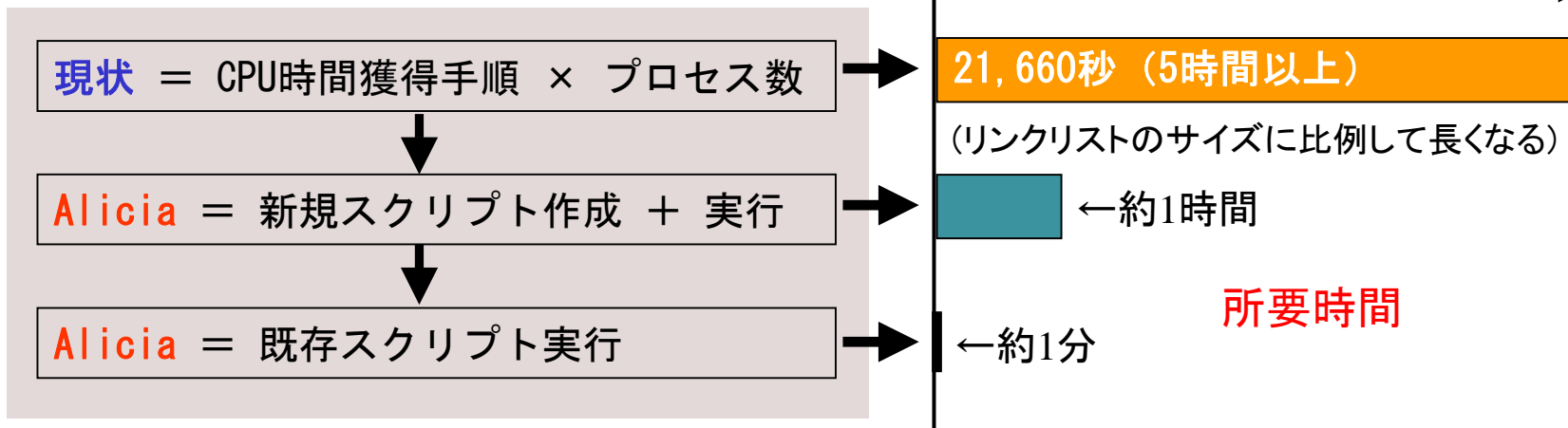
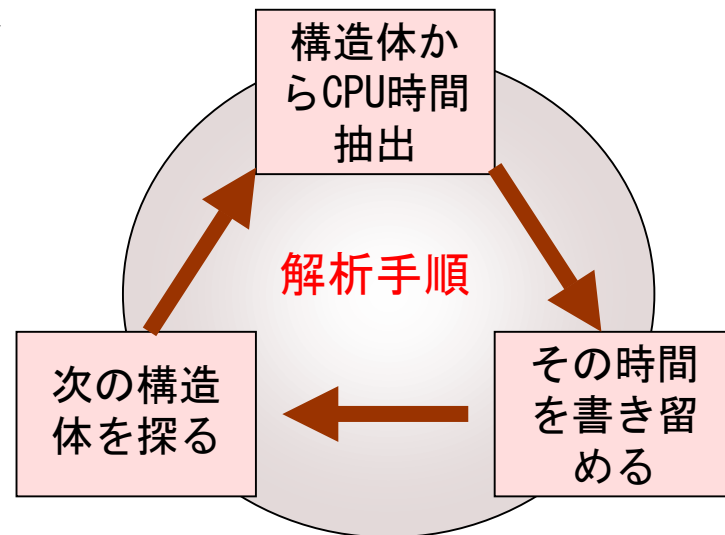
ダンプ解析者の要求：

今すぐ、task_struct構造体から全プロセスのCPU使用時間を抽出し、PSコマンドの出力に付加して表示したい！

- 実ダンプ採取環境：32CPUs, 4GB
- 実プロセス数 = 2166

推定所要時間：

- CPU時間獲得手順の時間 = 約10秒
- スクリプト作成時間 = 約1時間
- スクリプト実行時間 = 約1分





カーネル性能評価向けLKST(Linux Kernel State Tracer)機能追加

● 既存ツールと新規開発ツールの使い分けのノウハウ蓄積

- ・oprofile (<http://oprofile.sourceforge.net/>)
- ・hardmeter (<http://sourceforge.jp/projects/hardmeter/>)
- ・Lockmeter (<http://oss.sgi.com/projects/lockmeter>) など

● アプリケーション処理中のカーネル処理部分の性能評価

- (1) システムコールの開始から処理終了までの時間
- (2) IO要求から終了までの時間
- (3) ユーザプロセスの生成から終了までの時間、など
- (4) 上記処理中のカーネル処理時間の内訳
 - (a) メモリ確保に要する時間、実行回数
 - (b) ロックの取得に要する時間、実行回数
 - (c) プロセスの状態変化と各状態でのいる時間、など

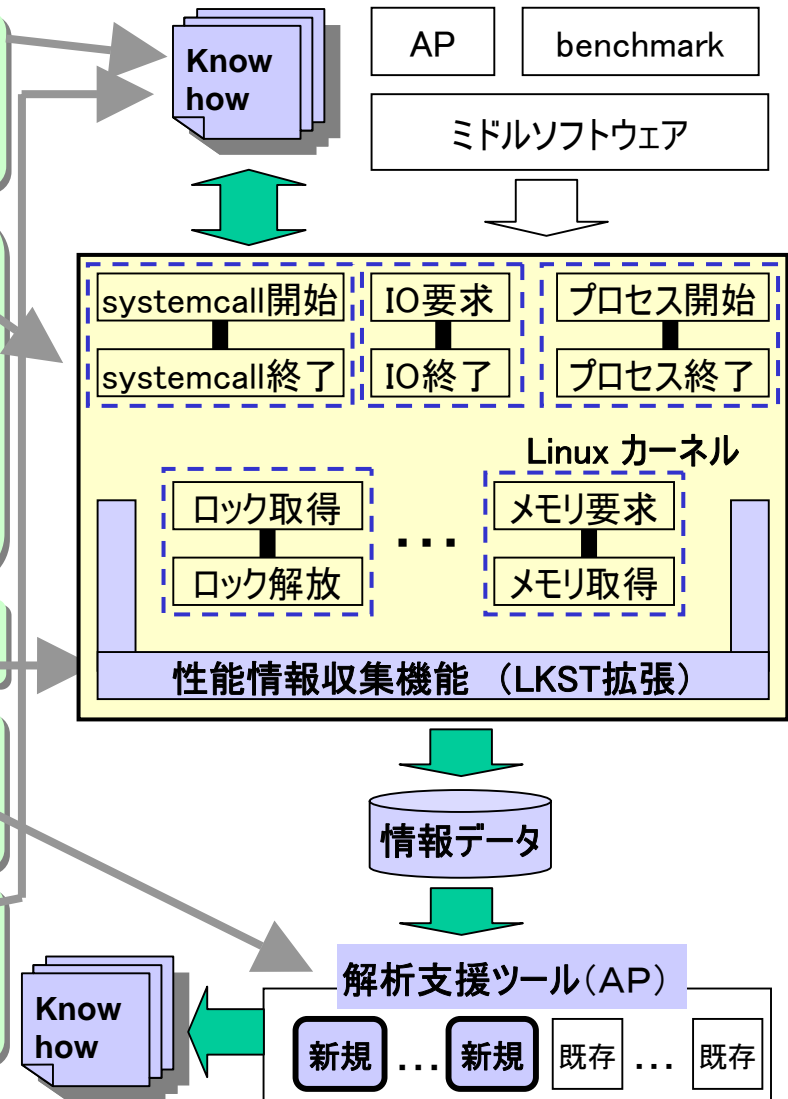
● 情報収集用にLKSTの情報収集部の機能を拡張

● 性能解析支援ツールの開発

- (a) 評価対象データの抽出と可視化
- (b) 状態変化データの抽出と可視化、など

● 性能解析ノウハウの蓄積

- (a) 評価目的に適した情報の収集方法の蓄積
- (b) 解析支援ツールの利用ノウハウの蓄積





2004年度開発結果の評価

●カーネル内部トレース情報を収集する機能をLKSTを拡張して実現

- ・ システムコール毎にかかる処理時間
- ・ メモリ確保・解放にかかる処理時間
- ・ デバイス毎、IO処理毎にかかる処理時間
- ・ スケジューリングアルゴリズムの処理時間

●収集したデータを分析・可視化表示するツールを開発

- ・ 各処理の最大、最小、平均値、処理の実行回数の算出
- ・ 上記の分布図やグラフを生成・表示

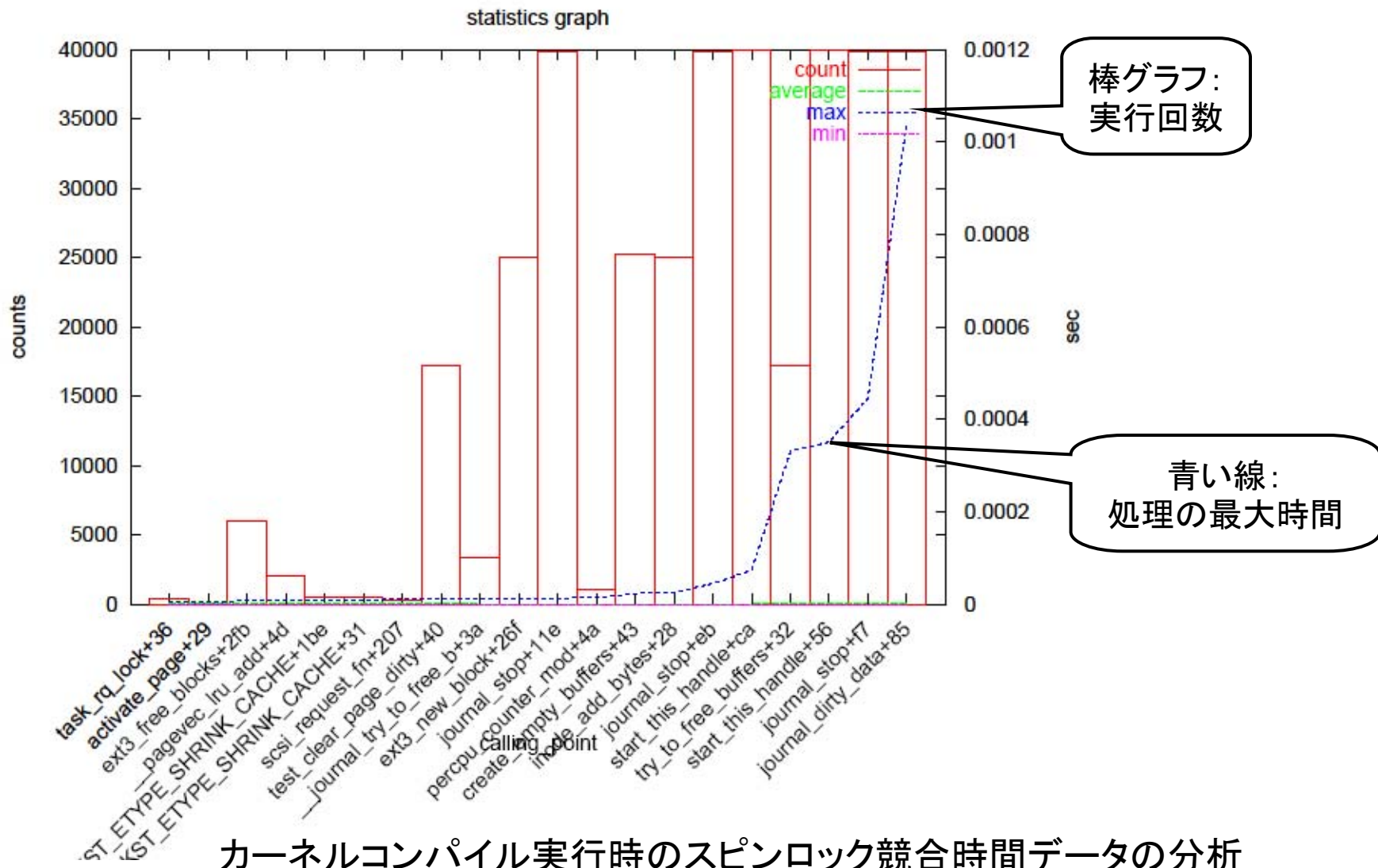
●カーネル内部処理の分析

- ・ 各システムコールの実行回数と最大処理時間、およびその実行回数内での処理時間分布を把握できた
⇒処理にとっても時間がかかるケースがあることを検出、ボトルネック解析のヒントとなる
- ・ カーネルのバージョンの違いによる、処理性能の傾向の違いが把握できた
⇒バージョンを変更する際の、性能変化を見積もるヒントとなる
- ・ ブロックIO処理毎の処理時間の傾向とキューの状態が把握できた
⇒ファイルIO時のボトルネック解析のヒントとなる

●評価結果グラフ:次紙参照

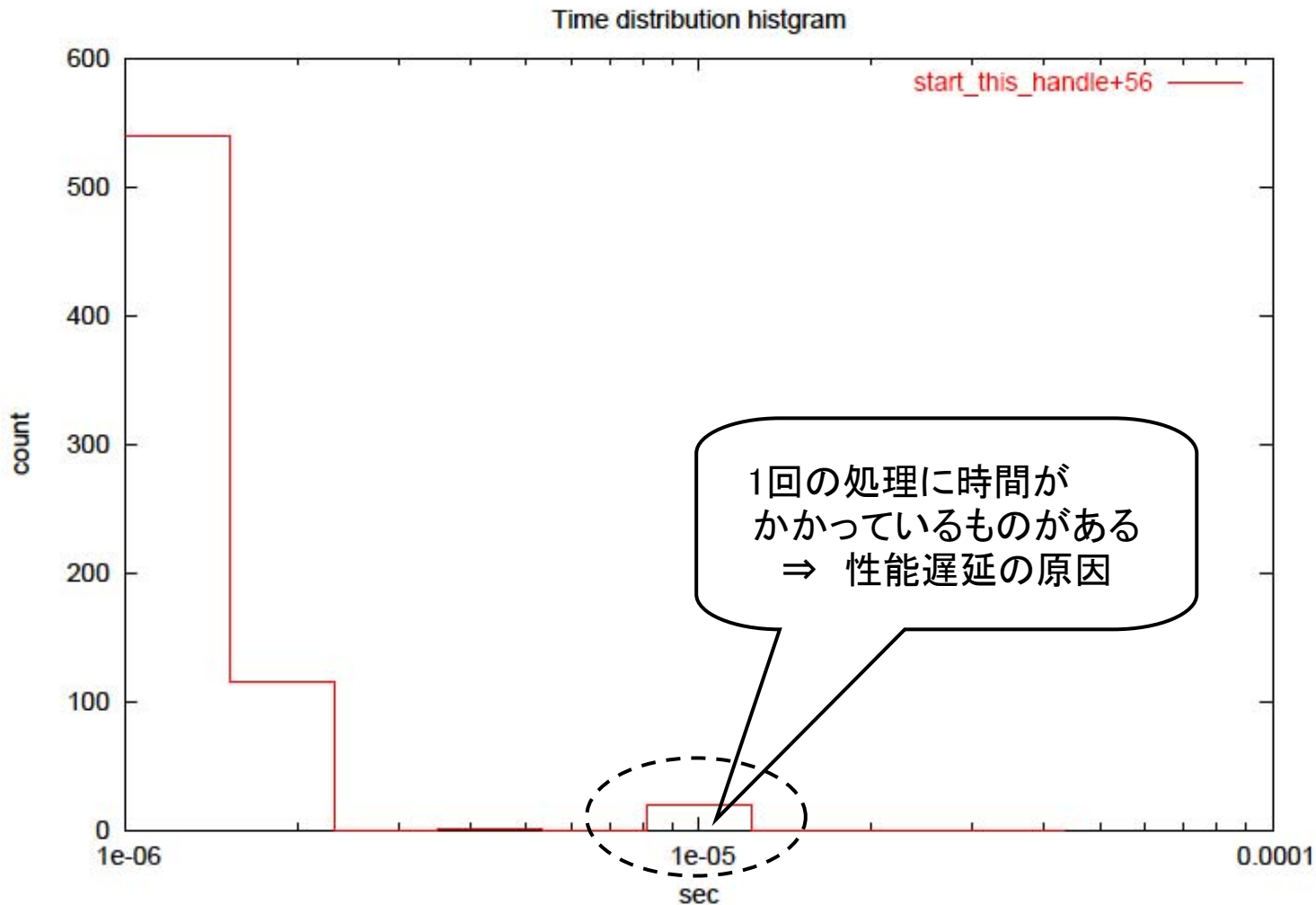


LKSTの出力結果





LKSTの出力結果



関数 start_this_handle+56 の性能データの分析



ディスク割り当て評価ツール

● ファイルシステムの情報取得: (新規開発)

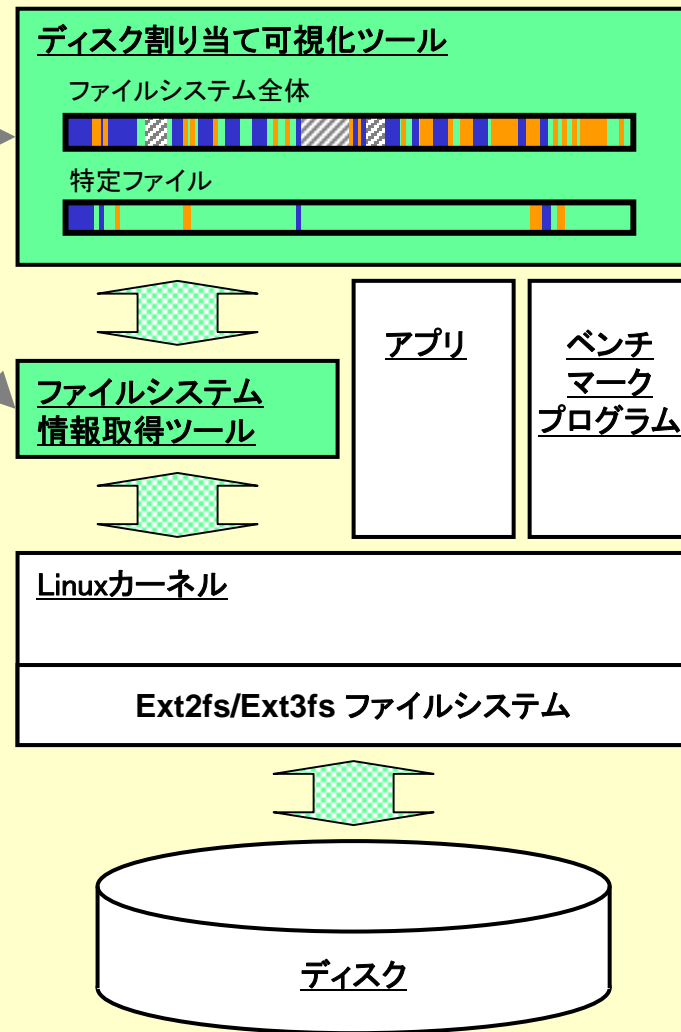
- 管理情報・構成の調査 (ext2fs, ext3fs)
- ファイルシステムの構成情報取得
 - ・使用ファイルシステムの種別取得
 - ・管理情報の取得 (パーティション構成、マウント有無)
- パーティションのディスク割り当て情報取得
 - ・マウントされていないパーティションの割り当て状況
 - ・マウントされているパーティションの割り当て状況

● ディスク割り当て評価解析: (新規開発)

- 各パーティションのファイルシステムのディスク割り当て状況の可視化
- 特定ファイルのディスク割り当て状況の可視化

● ノウハウ蓄積:

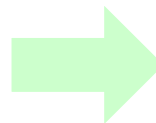
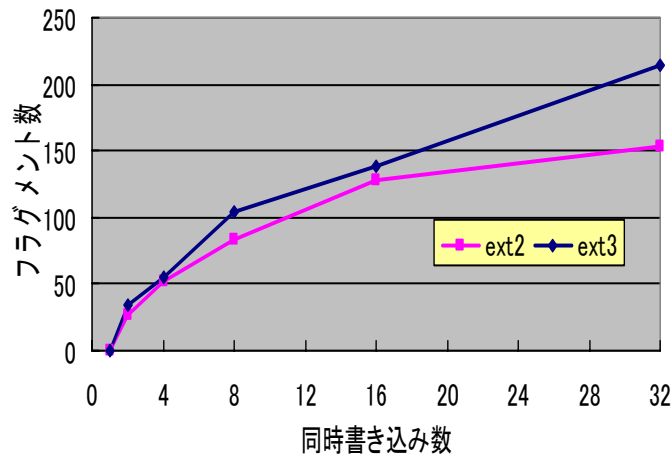
- ディスク割り当ての断片化の性能劣化への影響評価
- ディスク割り当て最適化ツールの評価
- ディスク割り当て性能劣化対策のノウハウ



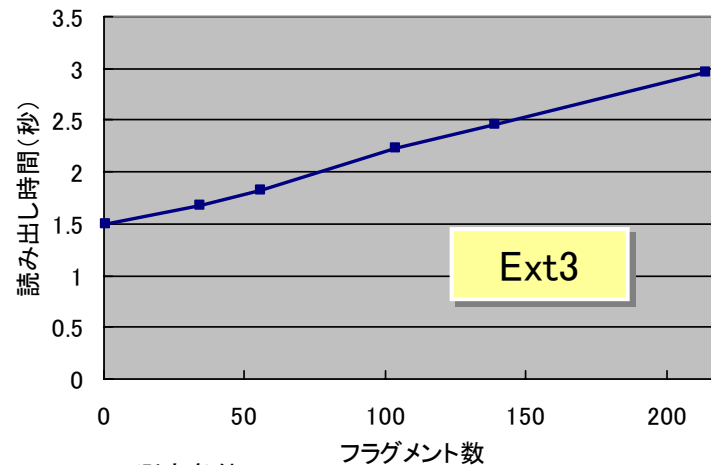
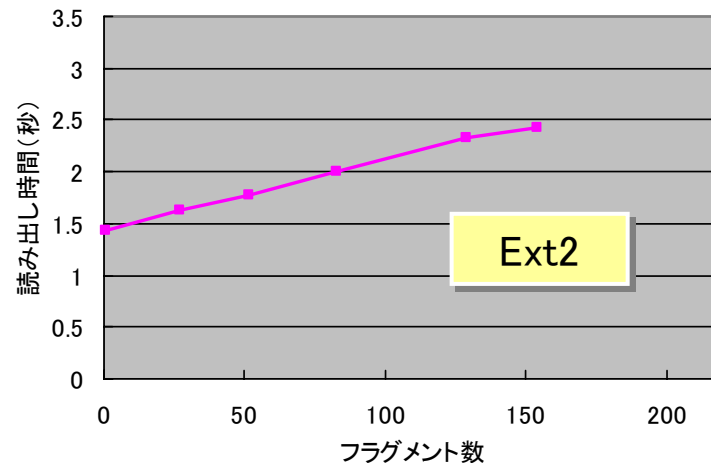


フラグメンテーションとファイルread性能の関係

ツールでフラグメンテーションを発生させる



ファイルのRead性能を測定



フラグメンテーション発生ツール:

- ・子プロセス毎に32MBのファイルを他パーティションにcopyする。子プロセスは合計32個。
- 同時書き込み数とはその子プロセスの同時実行数。
- ・上記の性能はcopy後のファイルをreadした性能。

実行結果:

- ・並列実行度が上がるにつれてフラグメンテーション数が増加する
- ・フラグメンテーション数が増加するとread性能が低下する。

測定条件:

- | | |
|--------------------|-------------------------|
| ●ハードウェア: | ●ソフトウェア: |
| ・CPU:P4 2GHz | ・OS - Miracle Linux 3.0 |
| ・メモリ:512MB | ・Filesystem - Ext2/Ext3 |
| ・HDD:30GB*2パーティション | (ブロックサイズ:4KB) |



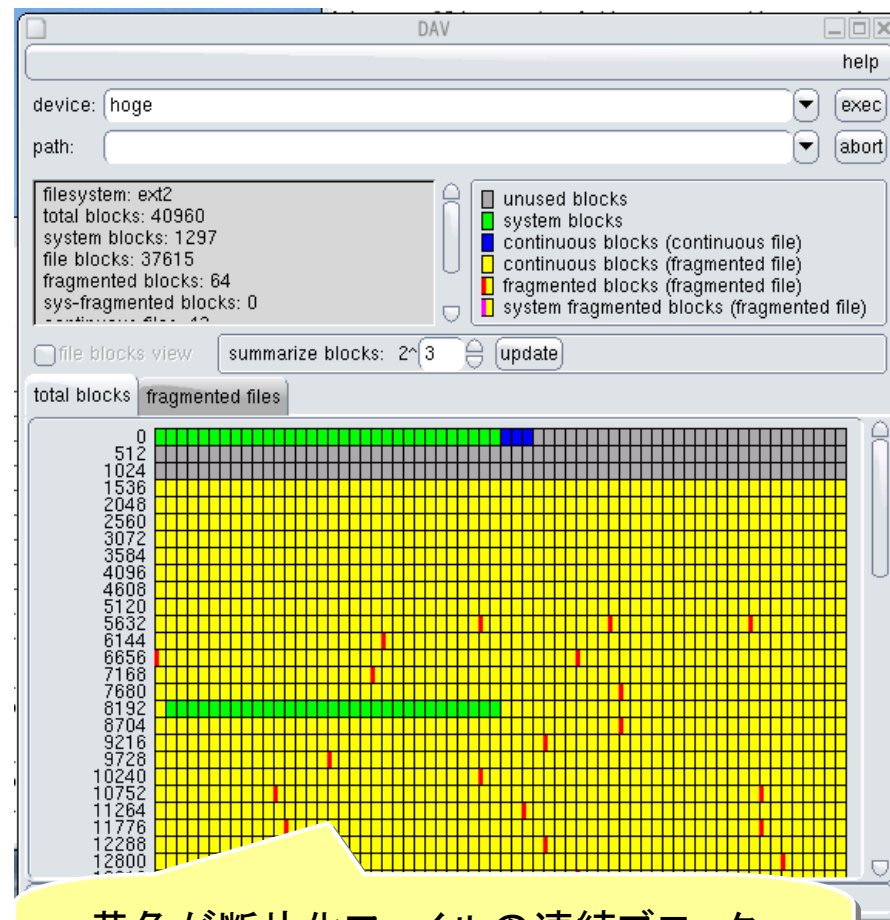
2004年度開発結果

- 性能劣化の原因やデフラグツールの効果を、視覚的に裏付け情報を取得する手段の提供 (右図)

- ・対象パーティション、ディレクトリ、ファイルのフラグメンテーション状態がどうなっているかの確認。
- ・デフラグツール適用前後における、Disk内の状況を把握可能。
- ・DAVはフラグメンテーション情報を収集するツール(dac)と情報を可視化するツール(dav)で構成

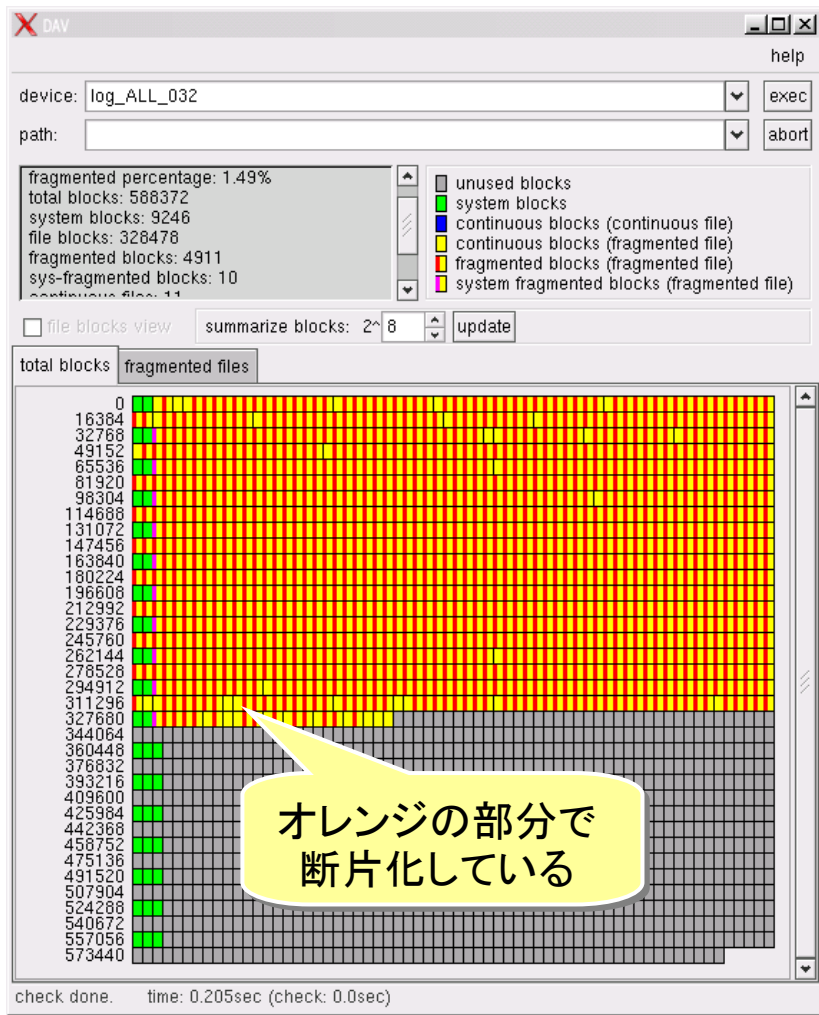
- Diskのフラグメント状態の遷移を取得する手段および視覚化する手段の提供

- ・dacで定期的に情報を保存
- ・保存したデータファイルを指定して、davを実行
- ⇒ フラグメンテーションがどう進んでいくのかを確認可能。
- 事前評価として行なうことで、サイジングに活用できる。
- 但し、本年度は手動で保存。



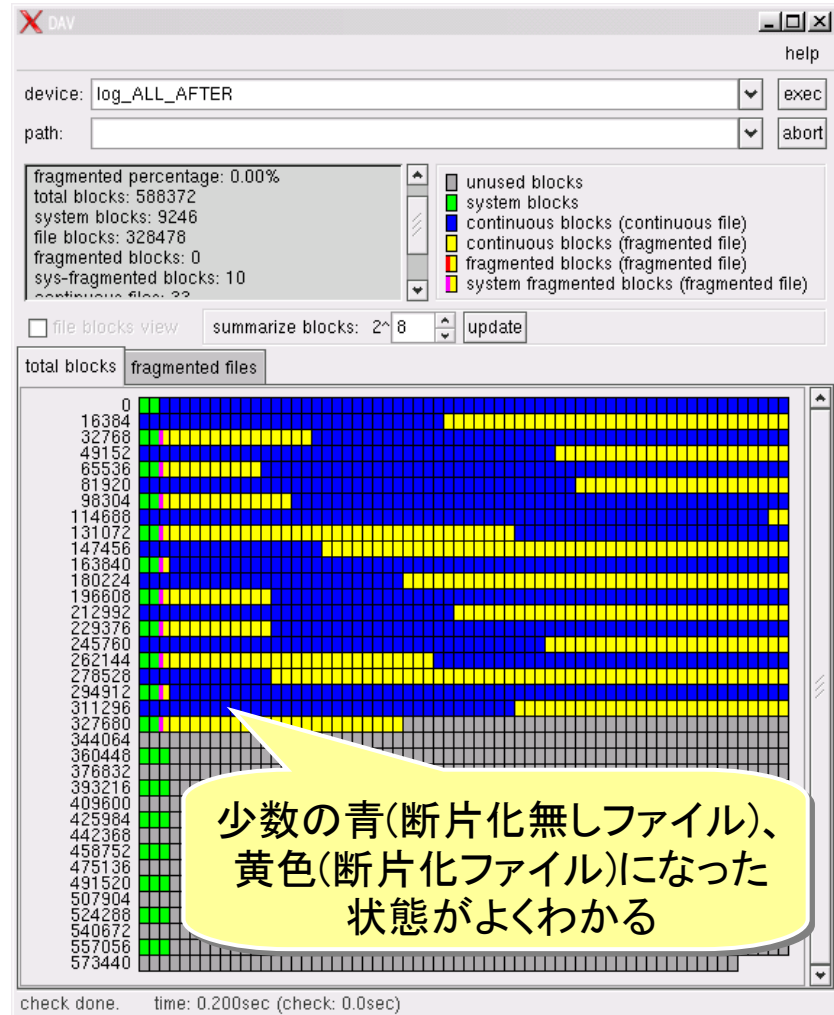
黄色が断片化ファイルの連続ブロック、
オレンジが断片化部分状態を表す。
(青は断片化していないファイル)

15. ディスク割当評価ツール (2) 2004年度開発結果評価③



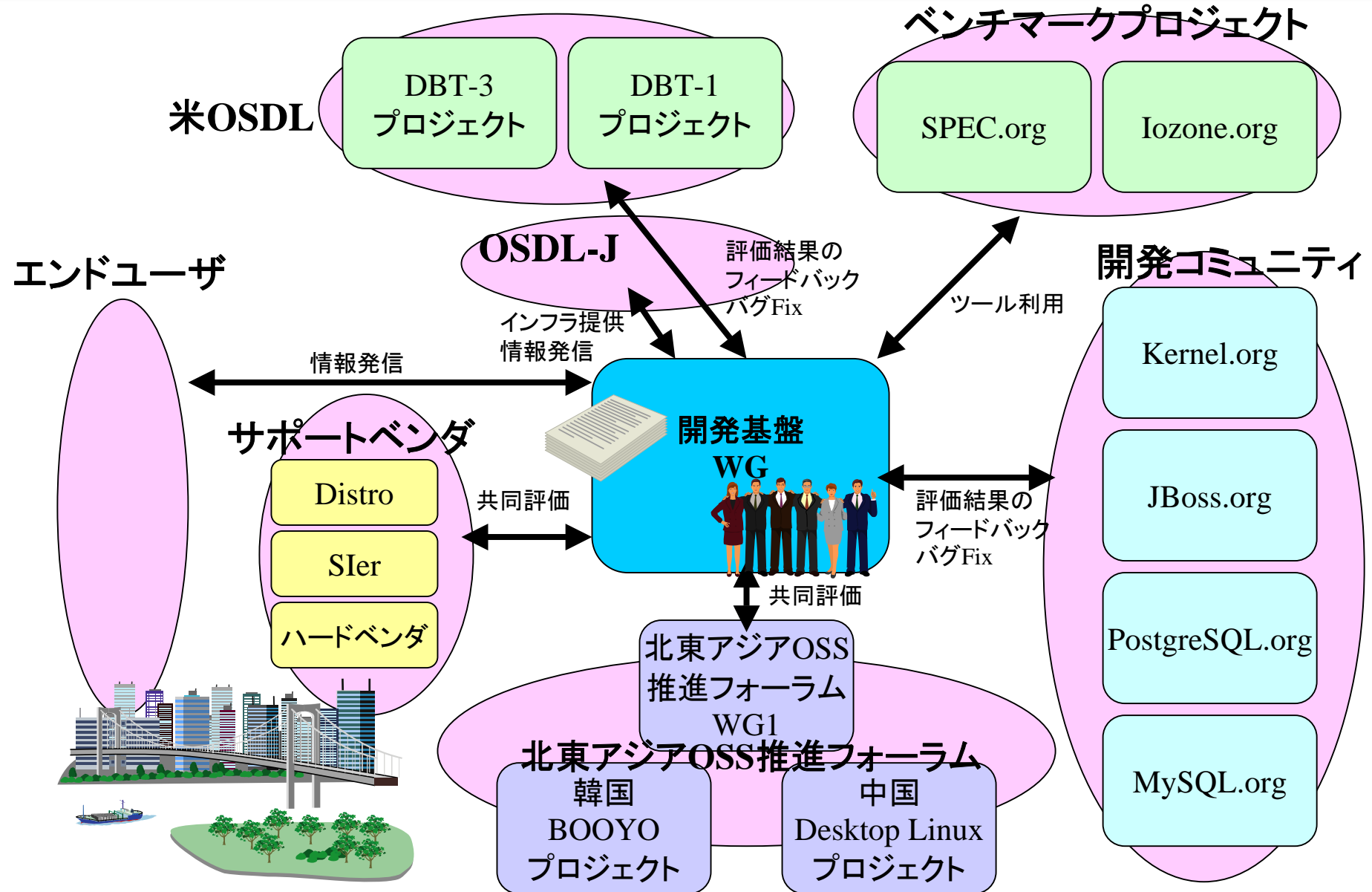
デフラグ実行前

デフラグツール: defrag 0.73 pjml



デフラグ実行後

16. 開発基盤WGとコミュニティとの関係





- Linux、OSSミドルは着実に進歩している。また、OSSの開発方式により、今後も発展が期待される
- これまでは明確な性能評価基準がなかったが、ベンダ共同で評価手順を策定し、また、これを共有し、育成していく土壌ができたことは大きな価値がある
- 開発コミュニティにとっても、1つの明確な評価基準が明らかになったことで、これを1つの評価尺度として利用しながら開発を進めることができるはず
- ユーザにとっては、自社のシステムにおいて、OSSがどこまで適用できるのかといった疑問に対し、1つの判断基準を提供できたと考える
- 今後は
 - ・評価対象のバリエーションを増やしていくこと
 - ・実際のシステム構築の現場での利用可否の検証をしていくこと
 - ・WorldWideに広く普及させるための活動を継続していくことを実施していきたい



発表資料: <http://www.ipa.go.jp/software/open/forum/>