SGI ProPack™ v 2.1.1 for Linux® Start Here（日本語版）

007-4558-002JP

制作協力
著作 : Julie Boney
編集 : Susan Wilkening
イラスト : Chrystie Danzer
制作 : Glen Traefald

表紙デザイン : Sarah Bolles (Sarah Bolles Design)、Dany Galgani (SGI Technical Publications)

改訂履歴

バージョン	説明
001	2003 年 2 月 初版発行
002	2003 年 2 月 SGI ProPack v2.1.1 for Linux リリース改版

目次

図一覧 ix

表一覧 xi

関連ドキュメント xiv

Intel コンパイラ関連文書 xiv

その他の Intel 関連文書 xiv

SGI 文書 xiv

SGI ドキュメントの入手先 xvi

本書の表記法 xvi

読者からのフィードバック xvii

1. リリースの内容 1

ソフトウェアの概要 1

CD の内容 2

インストールの概要 4

2. ソフトウェアの計画とインストール 5

ソフトウェアの計画 5

ディスク・パーティション表 6

ファイルの設定 6

SGI Linux Environment 7.2 ベース OS のインストール 6

SGI ProPack for Linux のインストール 10

SGI Linux Environment 7.2 updates のインストール 11

System Controller Software 1.1 CD のインストール 11

ソフトウェアのアップグレード 12

損傷した Root ファイルシステムの修復 12

Root ファイルシステムのマニュアル・マウント 14

3. サポートされる製品 17

OS 拡張 21

CpuMemSets のサポート. 21

Cpuset のサポート. 22

完全システム・アカウンティング (CSA:Comprehensive System Accounting) 22

システム・パーティション 23

I/O サブシステム 24

PCI-X バス固定ナンバリング. 24

Ethernet デバイスの固定命名 26

XSCSI サブシステム 27

XSCSI デバイス命名法 27

XFS ファイルシステム 28

XVM ボリューム・マネージャ 28

HPC アプリケーションのツールとサポート. 29

Message Passing Toolkit. 29

Performance Co-Pilot (PCP) 29

システム管理 30

PROM チップ 30

拡張ファームウェア・インターフェイス (EFI: Extensible Firmware Interface) 30

FLEXlm 32

SGIconsole 33

システム・コントローラ・ファームウェア 33

NUMA Tools 34

dlook コマンド 34

dplace コマンド 34

4. パフォーマンス・チューニング 35

システム構成の確認 35

シングルプロセッサ・コードのチューニング 36
出力の正確性の確認 37
ddd の使用 38
ヒープ破壊問題対策 39
チューニング済みコードの使用 40
チューニング必要部分の決定 41
コンパイラ・オプションの選択と適用 42
キャッシュ・パフォーマンスのチューニング 43
メモリ管理 43
chatr によるスタックとデータ・セグメントの変更 44
マルチプロセッサ・コードのチューニング 45
コードの並列化 45
コード中のボトルネックの検出 47
無効な共有の修復 47
dplace と *runon* の利用 48
MPI コードに対する *dplace* の使用 48
OpenMP コードに対する *dplace* の使用 48
環境変数によるパフォーマンス・チューニング 49
プロファイラとパフォーマンス・ツール 50
pfmon を使ったプロファイリング 50
profile.pl スクリプト 51
MPI プログラムでの *profile.pl* の使用 51
VTune を利用したリモート・サンプリング 52
GuideView の利用 52
その他のパフォーマンス・ツール 53

索引 55

図一覧

図 1-1 SGI ProPack for Linux Release CD の内容 3

表一覧

表 2-1 devfsディスク・パーティション 6

表 3-1 SGI ProPack v2.1.1 for Linux 製品 18

表 3-2 EFI コマンド 30

このマニュアルについて

このマニュアルは、SGI ProPack for Linux に関する情報を解説します。本書は次の各章から構成されます。

- 第 1 章「リリースの内容」では、このリリースの主な機能、CD の内容、および関連ドキュメントについて説明します。
- 第 2 章「ソフトウェアの計画とインストール」では、ディスクのパーティション設定、システムの利用を始める際に必要となる設定ファイルの作成と配置、SGI ProPack for Linux のインストールの手順についてフローチャート形式で説明します。
- 第 3 章「サポートされる製品」では、SGI LX 3000 システムでサポートされる製品コンポーネントを示します。
- 第 4 章「パフォーマンス・チューニング」では、シングルプロセッサ環境およびマルチプロセッサ環境でのプログラムのチューニングについて解説します。

本書の内容とその他の SGI ProPack for Linux ドキュメント、ディストリビューション CD に含まれる RPM に関するその他のドキュメントは CD "SGI ProPack v2.1.1 for Linux - Documentation CD" に入っています。このドキュメント CD の内容を見るには、Web ブラウザ・プログラムを使って `index.html` ファイルを開いてください。このオンライン・ファイルには本書作成後に記述された追加情報が含まれていることがありますので、目を通すようにしてください。

メモ：ソフトウェアおよびドキュメントに関するリリース時の最新情報は SGI ProPack for Linux Documentation CD のルート・ディレクトリにある `README.TXT` ファイルに格納されています。

関連ドキュメント

有用な情報が記載されたドキュメントと参考文献の一覧を次に示します。

Intel コンパイラ関連文書

Intel コンパイラに関する資料はシステム・ディスクのコンパイラがインストールされているディレクトリの /docs サブディレクトリにあります。Intel コンパイラのインストール後、下記の資料が参照可能です。

- 『Intel C++ Compiler User's Guide』(*c_ug_lnx.pdf*)
- 『Intel Fortran Compiler User's Guide』(*for_ug_lnx.pdf*)
- 『Intel Fortran Programmer's Reference』(*for_prg.pdf*)
- 『Intel Fortran Libraries Reference』(*for_lib.pdf*)

その他の Intel 関連文書

次の参考文献はItanium(旧名称IA-64)アーキテクチャやその関連事項について解説しています。

- 『Intel Itanium 2 Processor Reference Manual for Software Development and Optimization』
  (http://developer.intel.com/design/itanium/manuals からオンライン入手可)
- 『Intel Itanium Architecture Software Developer's Manual』
  (http://developer.intel.com/design/itanium/manuals からオンライン入手可)
- 『Introduction to Itanium Architecture』
  (http://shale.intel.com/softwarecollege/CourseDetails.asp?courseID=13 からオンライン入手可、要セキュア接続)

SGI 文書

SGI からは次の文書が入手可能です。

- 『Linux Device Driver Programmer's Guide』

デバイス・ドライバのプログラミング、構成、制御についての情報が記載されています。

- 『Message Passing Toolkit: MPI Programmer's Manual』
  SGI システム用に最適化された業界標準メッセージング・プロトコルについての解説です。
- 『Origin 2000 and Onyx2 Performance Tuning and Optimization Guide』
  MIPS/IRIX システムでのパフォーマンス・チューニングと最適化についての記述ですが、ハードウェアや OS に依存しない一般的なガイドラインについても述べられています。
- 『Performance Co-Pilot for Linux User's and Administrator's Guide』
  Performance Co-Pilot (PCP) ソフトウェア・パッケージの解説です。PCP は SGI システム用に開発されたパフォーマンス解析ツールで、Linux システム上でも動作します。
- 『Resource Administration Guide for Linux』
  Linux OS 搭載 SGI システム管理者向けのリファレンス・ガイドです。
- 『SGI LX 3000 User's Guide』
  SGI LX 3000 システムのアーキテクチャ概要と主なコンポーネントについての解説です。システムの電源投入とシャットダウンの基本操作、トラブルシューティング、設置と運用に関する安全基準などの情報も含まれます。
- 『SGI ProPack v2.1.1 for Linux Release Notes』
  ソフトウェアおよびドキュメントに関するリリース時の最新情報が納められています。この文書は SGI ProPack for Linux Documentation CD のルート・ディレクトリにある `README.TXT` ファイルに格納されています。
- 『SGIConsole 1.1 Start Here』
  SGIConsole の概要が解説されています。
- 『SGI L1 and L2 Controller Software User's Guide』
  システム・コンソールから実行する L1/L2 コントローラ・コマンドの解説書です。これらのコマンドは SGI LX 3000 Series システムのモニタリングと管理に使用します。
- 『XFS for Linux Administration』
  XFS についての解説です。XFS はオープン・ソースのジャーナル・ファイルシステムで、高速回復性、ダイレクト I/O、スペース・プリアロケーション、アクセス・コントロール・リスト、quotas など、業務用ファイルシステムに必要な機能を備えています。

SGI ドキュメントの入手先

SGI が公開するドキュメントは下記の方法で入手できます。

- SGI Technical Publications Library (http://docs.sgi.com) にアクセスする方法。ドキュメントは各種のフォーマットで提供されます。オンライン・ブック、リリース・ノート、マン・ページ、その他の文献の最新版が入手できます。
- マン・ページは、コマンドラインから「`man` <*title*>」と入力すると表示されます。

本書の表記法

本書では、次の表記が使用されます。

書式	内容
`command`	コマンド名、ファイル名、ルーチン名、パス名、シグナル、メッセージ、プログラミング・コードは等幅フォントで表示されます。
variable	変数、新出の用語は斜体で表示されます。
`user input`	用例中、ユーザが入力する文字は太字の等幅フォントで表示されます。
`manpage`**(***x***)**	マン・ページのタイトルは、かっこ内にセクション番号が表記されます。
[GUI 要素 **]**	ウィンドウ、画面、ダイアログ・ボックス、メニュー、ツールバー、アイコン名、ボタン、ボックス、フィールド、リストなどの GUI (Graphics User Interface) 要素名は、角かっこ [ ] で囲んで表示されます。

読者からのフィードバック

この文書の技術的な正確性、内容、または編成についてのご意見がありましたら、SGI までご連絡ください。ご意見とともに、マニュアルのタイトルと文書番号をお知らせください。(オンライン文書の文書番号は、文書の 1 ページ目に記載されています。印刷文書の文書番号は、本文ページ下端に記載されています。)

弊社には次のいずれかの方法でご連絡いただけます。

- 電子メールで連絡される場合は、下記アドレスをご利用ください。

  techpubs@sgi.com

- Technical Publications Library Web ページ(下記 URL)にアクセスされる場合は、Feedback 欄からコメントを送信してください。

  http://docs.sgi.com

- ご購入元のカスタマー・サービス担当にご連絡される場合は、問題点が SGI のバグ・トラッキング・システムにファイルされるようご相談ください。

- 書面でのお問合わせは、下記住所まで郵送ください。

  Technical Publications
  SGI
  1600 Amphitheatre Pkwy, M/S 535
  Mountain View, California 94043-1351 USA

- FAX で送信される場合は、+1 650 932 0801 “Technical Publications” 宛までお送りください。

お送りいただいたご意見には迅速に対応いたします。

リリースの内容

この章では SGI ProPack for Linux の概要と CD 内容の構成、ドキュメントに関する情報、およびインストール手順の概要について説明します。

ソフトウェアの概要

Linux はオープン・ソースと Linux コミュニティを特徴とする共同体制により開発されており、大規模サーバやスーパーコンピュータを提供するコンピュータ・メーカーに対して新しい開発モデルの方向性を示しています。大規模コンピュータ・システムに Linux OS を採用する利点は、より高度なソフトウェア保護、そしてメーカー製のオペレーティング・システムやハイ・パフォーマンス・コンピューティング・アプリケーション / コードと、サード・パーティやユーザによるソフトウェア開発との間で、より優れた統合化を実現できることです。また、Linux コミュニティによる OS の進化と企業メーカーによる開発が連動することで、企業単独によるオペレーティング・システム開発とは異なる方向から、革新と進歩の新たな可能性がもたらされます。

SGI ProPack for Linux 製品は、技術や芸術分野で要求される大規模なデータ処理と計算量のために Linux OS と Itanium プロセッサが適用される局面において理想的な処理機能とパフォーマンスを備えています。SGI ProPack for Linux は Red Hat 7.2 をベース OS として、各種の機能が追加、拡張されています。SGI ProPack for Linux は SGI LX 3000 Series の任意の構成で動作するよう設計されています。このリリースでサポートされる SGI ハードウェア・プラットフォームおよび OS 構成の範囲は、下記 URL から入手できるドキュメントに詳しく説明されています。

```
http://support.sgi.com/linux
```

CD の内容

SGI ProPack for Linuxには、次のCDが含まれています（各CDの内容は図1-1に示されています）。

- SGI Linux Environment 7.2 Installation CD セット（2 枚）
- SGI Linux Environment 7.2 Updates CD
- SGI Linux Environment 7.2 Source Code CD セット（3 枚）
- SGI ProPack v2.1.1 for Linux Boot CD
- SGI ProPack v2.1.1 for Linux Open/Free Source Software CD
- SGI ProPack v2.1.1 for Linux Proprietary Software CD

メモ：この CD には SGI システム専用ソフトウェアが含まれています。再配布は認められません。パッケージに印刷されているライセンス条項を参照してください。

- SGI ProPack v2.1.1 for Linux Documentation CD
- System Controller Software 1.1 CD (for IRIX and Linux)

メモ：このソフトウェアは Origin / Onyx 3000 Series システム（MIPS プロセッサ・ベースの SGI システム）、および SGI LX 3000 Series システムに対してのみライセンスされます。この CD は SGI ライセンス条項と使用制限に従って使用する必要のあるソフトウェアを含みます。この CD の SGI ライセンス条項では、CD 内容が SGI システムにのみインストール可能であるということが規定されています。

SGI Linux Environment7.2 および SGI ProPack v2.1.1 for Linux CD の内容の一覧は 3 ページの図 1-1 を参照してください。ソフトウェアとドキュメントに関するリリース時の最新情報は SGI ProPack for Linux Documentation CD のルート・ディレクトリにある `README.TXT` ファイルに格納されています。

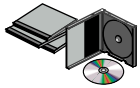
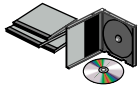
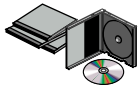
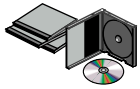
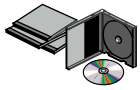
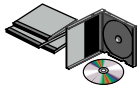
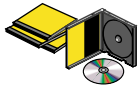
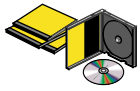
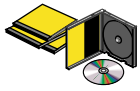
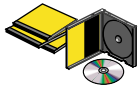
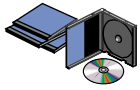
* SGI Linux Environment 7.2	
 SGI Linux Environment 7.2 Installation CD 1 of 2 CD1	インストーラ本体
 SGI Linux Environment 7.2 Installation CD 2 of 2 CD2	
 SGI Linux Environment 7.2 Updates CD3	Red Hat 7.2 ソフトウェアのアップデータ
 SGI Linux Environment 7.2 Source Code CD 1 of 3 CD4	ベース OS とその他のオープンソース・アプリケーションのソースコード
 SGI Linux Environment 7.2 Source Code CD 2 of 3 CD5	
 SGI Linux Environment 7.2 Source Code CD 3 of 3 CD5	
* SGI ProPack v2.1.1 for Linux	
 SGI ProPack v2.1.1 for Linux Boot CD CD1	デバイス・ドライバと boot カーネル
 SGI ProPack v2.1.1 for Linux Open Source Software CD2	GPL/LGPL ライセンスおよびその他のオープンソース・ソフトウェア、最新 SGI プラットフォームと NUMA サポート、カーネルの更新、CpuMemSets とパフォーマンス評価ツール、XSCSI 環境、QL-SCSI、XFS、CSA、LKCD、kdb
 SGI ProPack v2.1.1 for Linux Proprietary Software CD3	システム PROM、XVM、cpuset、numatools（dlock/dplace など）、Array Services、FLEXlm、システム・パーティション・ソフトウェア、MPT
 SGI ProPack v2.1.1 for Linux Documentation CD CD4	ドキュメント・セット、FAQ、HOWTO、マン・ページ
* System Controller Software 1.1 CD	
 System Controller Software 1.1 CD	システム・コントローラ・ソフトウェアおよびファームウェア（IRIX / Linux 用）

図 1-1 SGI ProPack for Linux Release CD の内容

SGI のオープン・ソースに関連する開発計画の詳細は、次の Web サイトで公開されています。

http://oss.sgi.com

その他のオープン・ソース（LKCD など）情報は、次の Web サイトで公開されています。

http://sourceforge.net

インストールの概要

通常、SGI ProPack for Linux は SGI プラットフォームにプリインストールされて出荷されます。再インストールが必要な場合は、まず SGI ProPack v2.1.1 Boot CD (CD1) からブートして、次に SGI Linux Environment 7.2 Installation CD セット (CD1/CD2)、SGI ProPack v2.1.1 Open/Free Source Software (CD2)、および Proprietary Software (CD3)、そして最後に SGI Linux Environment 7.2 Updates (CD3) の順でインストールしてください。インストールの詳細は第 2 章「ソフトウェアの計画とインストール」に説明されています。

ソフトウェアの計画とインストール

この章では SGI ProPack for Linux のインストールに必要な計画と手順について説明します。

SGI LX 3000 システムには、ベース Linux ディストリビューション (SGI Linux Environment 7.2) と SGI ProPack ソフトウェアがプリインストールされています。この章では、これらのソフトウェアを再インストールする必要が生じたときに CD からインストールする手順について説明します。

セキュリティ強化のため、Linux で root ログインするにはパスワードが必要です。プリインストールされているソフトウェアの場合、デフォルトのパスワードは `sgisgi` です。ログイン後、このパスワードは別のパスワードに置換えてください。

SGI ProPack ソフトウェアは SGI Linux Environment 7.2 ディストリビューションと共にインストールした場合のみ動作します。これより古いディストリビューションや他社のディストリビューションは、SGI ProPack ソフトウェアと共に動作しません。

システムのインストールと設定を始める前に、第 1 章「リリースの内容」を読み、SGI ProPack for Linux ソフトウェアの機能と構成の概要を理解してください。xiv ページの「関連ドキュメント」に示される書籍も参考になります。

ソフトウェアの計画

ここでは、同梱されるソフトウェアについて知っておく必要のあるディスクとファイルの情報を説明します。

ディスク・パーティション表

SGI から出荷されるシステムは、表 2-1 に示されるレイアウトでパーティションが設定されています。

表 2-1 devfs ディスク・パーティション

デバイス	マウント・ポイント	サイズ
/dev/xscsi/pci01.03.0-1/target1/lun0/part1	*/boot/efi*	500 MB
/dev/xscsi/pci01.03.0-1/target1/lun0/part2	swap	9 GB
/dev/xscsi/pci01.03.0-1/target1/lun0/part3	*/*	25 GB

デバイスの名前については 24 ページの「PCI-X バス固定ナンバリング」と 27 ページの「XSCSI デバイス命名法」も参照してください。

ファイルの設定

LKCD (Linux Kernel Crash Dump) は */var/dump* にファイルを作成します。*/var* ディレクトリのサイズは Performance Co-Pilot (PCP) と CSA (Comprehensive System Accounting) がそれぞれ自分のサブディレクトリにログを作成するにつれて増えます。通常、これらのファイルは 100 MB を越えません。LKCD のデフォルト設定の詳細は *lkcd_config* (1) マン・ページを参照してください。

メモ：lkcd ダンプが生成されると、lkcd は /var ディレクトリをすぐに数 GB 以上消費してしまいます。デフォルトの root ファイルシステムが比較的大きく設定されているのはこのためです。必要に応じて */var/dump* を独立させたり、lkcd の設定を変えることができます。

SGI Linux Environment 7.2 ベース OS のインストール

ここでは、SGI ProPack v2.1.1 for Linux Boot CD (CD1) と SGI Linux Environment 7.2 CD セットのインストール手順について説明します。

メモ：インストール画面には、前のステップに戻るためのボタンとインストールを中断するためのボタンがあります。これらのボタンを使用するには、Tab キーを押して選択したいボタンをハイライト表示させ、ENTER キーを押します。

1. Installation Boot CD (CD1) をシステムの CD-ROM ドライブにセットし、システムを再起動します。
2. システム起動メッセージの表示中に、デバイス・マッピングを確認します。次のようなメッセージ出力をチェックしてください。

```
Device mapping table

fs0  : Pci(2|1)/Ata(Primary, Master)/CDROM(Entry1)（この行が CD-ROM）
fs1  : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part1,Sig00000000)
fs2  : Pci(1|1)/Scsi(Pun0/Lun2)/HD(Part1,Sigg1)
blk0 : Pci(2|1)/Ata(Primary, Master)
blk1 : Pci(2|1)/Ata(Primary, Master)/CDROM(Entry1)
blk2 : Pci(1|1)/Scsi(Pun0/Lun1)
blk3 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part1,Sig00000000)
blk4 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part2,Sig00000000)
blk5 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part3,Sig00000000)
blk6 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part4,Sig00000000)
blk7 : Pci(1|1)/Scsi(Pun0/Lun2)
blk8 : Pci(1|1)/Scsi(Pun0/Lun2)/HD(Part1,Sigg10)
```

3. **Shell>** プロンプトが表示されたら、次のように CD-ROM デバイス名を指定します。

```
Shell> fs0:  （CD-ROM デバイス名を指定して移動）
fs0:\>
```

4. 「**elilo**」と入力して ENTER キーを押し、CD からブートします。ELILO boot: プロンプトが表示されたら、再度 ENTER キーを押します。

```
fs0:\> elilo （ブートローダを指定）
LoadPe: using PE image entry point
LoadPe: using PE image entry point
ELILO boot: （ENTER キーを押して Linux カーネルを起動）
```

5. CD がブートされ、インストールが始まります。
6. インストールで使用する言語と、システムのデフォルトとして設定する言語を選びます。**OK** を選択して次に進みます。

メモ：現時点では、コンソールでのアジア言語はサポートされません。SGI では、アジア系言語で使用するシステムのデフォルト・システム言語として英語 (English) を選択することを推奨します。

7. 次の画面では、インストール手法を選択できます。現時点で有効な選択肢は [Local CDROM] のみです。**OK** を選択して次に進みます。
8. SGI ProPack 情報画面が表示されます。**OK** を選択して次に進みます。
9. 次の画面ではマウスの有無を選択します。SGI LX 3000 システムでは [None] を選択します。**OK** を選択して次に進みます。
10. インストールのタイプが尋ねられます。[Custom] をリストから選択してください。**OK**. を選択して次に進みます。
11. パーティション画面が表示されたら、[Autopartition] を選択して、**OK** を選択します。警告メッセージが表示されたら [IGNORE] を選択します。
12. パーティション方法を尋ねられるので、[Remove all partitions on this system] を選択します。ブート・デバイスはデフォルトで選択されます。**OK** を選択して次に進みます。
13. デバイスのデータ消去に関する警告メッセージが表示されたら、[Yes] を選択してパーティションのレイアウトを実行することを承認します。
14. 次の画面では、必要に応じてパーティションのレイアウトを変更できます。パーティションを変更した場合は、`/boot/efi` ファイルシステム・パーティションのタイプが `vfat` に、スワップ・パーティションのタイプが `swap` に、その他のファイルシステムが `xfs` に、それぞれ設定されていることを確認します。**OK** を選択してパーティションのレイアウトを確定します。
15. ネットワーク・インターフェイスの設定に *`Bootp`* または DHCP を使用して次に進むにはそのまま **OK** を選択します。スタティック IP を使用する場合は、必要なパラメータを入力してから **OK** を選択して次に進みます。

メモ：IX ブリックのバス 1: スロット 2（IO9 のすぐ右）に Ethernet ボードを追加する場合は、26 ページの「Ethernet デバイスの固定命名」を参照してください（このようなボード追加はほとんどの場合必要ありません）。

16. 前の画面でスタティック IP を選択した場合は、システムのホスト名を尋ねられます。ホスト名を入力し、**OK** を選択して次に進みます。
17. 次の画面では、ファイアウォールの設定が必要かどうかが尋ねられます。適切なオプションを選択し、**OK** を選択して次に進みます。
18. 次の画面では、システムがサポートする追加言語を選択できます。

メモ：追加言語を選択する場合、デフォルト言語として使用する言語も指定する必要があります。デフォルト言語はインストール完了後、システムで使用されるようになります。デフォルト言語はインストール後も変更できます。現時点では、コンソールでのアジア言語はサポートされません。SGI では、アジア系言語で使用するシステムのデフォルト・システム言語として英語 (English) を選択することを推奨します。

---

19. 次の画面ではシステムのタイム・ゾーンを設定します。適切な情報を指定して、**OK** を選択して次に進みます。
20. 次の画面では、システムの root パスワードが尋ねられます。root ログインに使用するパスワードを入力して、**OK** を選択します。
21. 次の画面ではシステムのユーザを作成します。システムにログインする `root` 以外のユーザを設定します。ユーザを作成する場合はここで各欄に必要な情報を入力して、**OK** を選択します。すべて空欄のまま **OK** を選択すると、ユーザを作成せずに次に進みます。
22. 前の画面でユーザを作成した場合は、次のユーザを作成する画面が表示されます。必要に応じてユーザを追加していきます。**OK** を選択して次に進みます。
23. 次の画面では、認証の設定を行います。**OK** を選択して次に進みます。
24. [Package Group Selection] 画面が表示されます。ここでは、表示されるすべてのパッケージをインストール対象として選択する必要があります。[Everything]（リストの下端）を選択してください。`Tab` キーで **OK** をハイライト表示し、ENTER キーを押して次に進みます。
25. [Installation to begin] 画面が確認のために表示されます。**OK** を選択して次に進みます。
26. 次の数画面でパッケージのインストール状況が確認され、必要な時点で残りの CD をセットするよう求められます。CD をセットしたら **OK** を選択して、次に進みます。
27. インストールが完了したら **OK** を選択して、システムを再起動します。CD はドライブから取り出せます。

SGI ProPack for Linux のインストール

ここでは、SGI ProPack v2.1.1 for Linux Open/Free Source Software (CD2) と SGI ProPack v2.1.1 for Linux Proprietary Software (CD3) のインストール手順を説明します。

1. システムに `root` でログインします。初期インストールで指定したパスワードを入力します。
2. SGI ProPack Open/Free Source Software CD (CD2) をシステムにセットして、次のコマンドを入力して CD をマウントします。

   ```
   mount /dev/cdrom /mnt/cdrom
   ```

3. 「**`/mnt/cdrom/INSTALL`**」と入力します。
4. SGI ProPack の [Welcome] 画面が表示されます。**OK** を選択して次に進みます。
5. 次の画面でインストールのタイプを選択します。[Yes] を選択してから、**OK** を選択します。
6. [Package Group Selection] 画面が表示されます。この画面では、インストールするパッケージ・グループが選択できます。上下矢印キーを押して選択行を移動し、スペースバーで選択します。パッケージ・グループを選択後 `Tab` キーを押して **OK** をハイライト表示し、`ENTER` キーを押すと、選択したパッケージ・グループに対応する RPM がインストールされます。デフォルトではすべてのパッケージが選択されています。選択を変更した場合は、[SGI Proprietary] オプションが選択されていることを確認してください。**OK** を選択して次に進みます。
7. [Installation to Begin] 画面が表示され、インストールのログが *`/tmp/sgi-install.log`* に保存されることが示されます。`Tab` キーを押して **OK** をハイライト表示し、`ENTER` キーを押してください。
8. インストールが開始されます。[Package Installation] 画面にインストール中のパッケージ名とインストール経過時間が表示されます。
9. インストールが完了すると [Complete] 画面が表示されます。`ENTER` キーを押してください。`root` のプロンプトに戻ります。
10. インストール完了後、新規インストールされた SGI ProPack for Linux software カーネルを実行するためにシステムを再起動してください。「`reboot`」と入力して `ENTER` キーを押すと、システムを再起動できます。CD はドライブから取り出せます。

SGI Linux Environment 7.2 updates のインストール

ここでは、SGI Linux Environment 7.2 updates を含む CD (CD3) のインストールを説明します。

1. システムに root でログインします。初期インストールで指定したパスワードを入力します。
2. SGI Linux Environment 7.2 Updates CD をシステムにセットして、次のコマンドを入力して CD をマウントします。

```
mount /dev/cdrom /mnt/cdrom
```

3. 「**/mnt/cdrom/INSTALL**」と入力します。
4. [Welcome] 画面が表示されます。**OK** を選択して次に進みます。
5. 次の画面ではインストールのタイプが選択できます。[Upgrade Existing System] がデフォルトです。**OK** を選択して次に進みます。
6. 次の画面でアップグレードを開始します。アップグレードを始めるには **OK** を選択します。
7. 次の画面でインストールの進行状況が確認できます。
8. 次の画面で、SGI Linux Environment 7.2 update のインストール完了が確認できます。「reboot」と入力して ENTER キーを押すと、システムを再起動できます。CD はドライブから取り出せます。

System Controller Software 1.1 CD のインストール

ここでは、System Controller Software 1.1 (for IRIX and Linux) のインストール手順を説明します。このソフトウェアは MIPS プロセッサ・ベースの SGI システム Origin / Onyx 3000 Series および SGI LX 3000 システムに対してのみライセンスされます。

1. システムに root でログインします。初期インストールで指定したパスワードを入力します。
2. System Controller Software 1.1 CD をシステムにセットして、次のコマンドを入力して CD をマウントします。

```
mount /dev/cdrom /mnt/cdrom
```

3. /mnt/cdrom/RPMS/ia64 ディレクトリに移動して、次のコマンドを入力します。

   ./install

4. インストールが開始されます。インストール中のパッケージ名が画面に表示されます。インストール完了後、root のプロンプトに戻ります。
5. インストールが完了したら System Controller Software 1.1 CD を取り出します。

ソフトウェアのアップグレード

SGI ProPack ソフトウェアのアップグレードが必要になったら、新しい SGI ProPack CD セットを使って、本書 10 ページの「SGI ProPack for Linux のインストール」の手順を実行してください。

システム上の全ソフトウェアを再インストールする場合は、最新の SGI ProPack CD セットを使って、この章で説明されるインストール手順すべてを繰返してください。

損傷した Root ファイルシステムの修復

Root ファイルシステムが損傷した場合は、次の手順で修復を試みてください。

1. Installation Boot CD (CD1) をシステムの CD-ROM ドライブにセットし、システムを再起動します。
2. システム起動時に、デバイス・マッピング・テーブルを確認します。次のような出力を確認してください。

```
Device mapping table

fs0  : Pci(2|1)/Ata(Primary, Master)/CDROM(Entry1)（この行が CD-ROM）
fs1  : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part1,Sig00000000)
fs2  : Pci(1|1)/Scsi(Pun0/Lun2)/HD(Part1,Sigg1)
blk0 : Pci(2|1)/Ata(Primary, Master)
blk1 : Pci(2|1)/Ata(Primary, Master)/CDROM(Entry1)
blk2 : Pci(1|1)/Scsi(Pun0/Lun1)
blk3 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part1,Sig00000000)
blk4 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part2,Sig00000000)
blk5 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part3,Sig00000000)
blk6 : Pci(1|1)/Scsi(Pun0/Lun1)/HD(Part4,Sig00000000)
blk7 : Pci(1|1)/Scsi(Pun0/Lun2)
blk8 : Pci(1|1)/Scsi(Pun0/Lun2)/HD(Part1,Sigg10)
```

3. **Shell>** プロンプトが表示されたら、次のように CD-ROM デバイス名を指定します。

```
Shell> fs0:  （CD-ROM デバイス名を指定して移動）
fs0:\>
```

4. 次の行に示すように「**elilo linux rescue**」と入力します。

```
fs0:\> elilo linux rescue（ブートローダを起動）
```

5. CD がブートされ、レスキュー・モードが開始します。
6. このレスキュー・プロセスで使用する言語を選びます。**OK** を選択して次に進みます。

メモ：現時点では、コンソールでのアジア言語はサポートされません。SGI では、アジア系言語で使用するシステムの言語として英語 (English) を選択することを推奨します。

---

7. 次の画面では、レスキュー用イメージがあるメディアを選択できます。現時点で有効な選択肢は [Local CDROM] のみです。**OK** を選択して次に進みます。
8. 次の画面では、`root` ドライブをマウントするオプションが選択できます。ファイルシステムが損傷を受けている場合は、`xfs_repair` コマンドを実行する前にマウントを試みることが有効な場合があります。これによって、重要なデータをジャーナルから復元できる可能性があります。

   [Continue] を選択してインストーラがハングアップしたりクラッシュした場合は、再起動の後、ここまでのステップをやり直し、[Skip] を選択してください。その後、ファイルシステムをマニュアルでマウントします。この章の「Root ファイルシステムのマニュアル・マウント」を参照してください。
9. これまでの手順が成功した場合は、システムが `/mnt/sysimage` に検出されたというメッセージが次の画面に表示されます。**OK** を選択してレスキュー・モードのシェルに入ります。
10. この時点で、管理コマンドを使用して破損・削除されたソフトウェアの修復や再ロードが行えます。ここでは、破損した root ファイルシステムを修復する例を示します。

11. ジャーナルからデータを復元するためなどの理由でファイルシステムをマウントしている場合は、xfs_repair コマンドを実行する前にファイルシステムをアンマウントする必要があります。引数を指定せずに mount コマンドを実行すると、現在マウントされているファイルシステムの一覧が表示されます。例を次に示します。

    sh-2.05# mount

```
rootfs on / type rootfs (rw)
devfs on /dev type devfs (rw)
/dev/root.old on / type ext2 (rw)
none on /dev type devfs (rw)
/proc on /proc type proc (rw)
/dev/pts on /dev/pts type devpts (rw)
/tmp/cdrom on /mnt/source type iso9660 (ro)
/tmp/loop0 on /mnt/runtime type cramfs (ro)
/dev/xscsi/pci01.03.0-1/target1/lun0/part3 on /mnt/sysimage type xfs (rw)
/dev/xscsi/pci01.03.0-1/target1/lun0/part1 on /mnt/sysimage/boot/efi type vfat (rw)
none on /mnt/sysimage/dev/pts type devpts (rw)
none on /mnt/sysimage/proc type proc (rw)
```

    /mnt/sysimage type xfs (rw) で終わる行に、root ディスクに対応するデバイスが表示されています。上の例では /dev/xscsi/pci01.03.0-1/target1/lun0/part3 です。

12. 次のコマンドを実行してファイルシステムをアンマウントします。

    umount /mnt/sysimage

13. ファイルシステムに対して xfs_repair を実行します。例を次に示します。

    xfs_repair /dev/xscsi/pci01.03.0-1/target1/lun0/part3

14. xfs_repair が完了したらファイルシステムは修復されています。システムを再起動してください。

Root ファイルシステムのマニュアル・マウント

インストーラが root ファイルシステムを検出できない場合は、マニュアルでマウントする必要があります。ここではこの方法について説明します。詳細は『XFS for Linux Administration』を参照してください。まれに、破損したファイルシステムをマウントしようとすると mount コマンドがハングアップしたりトラブルを起こすことがあります。この場合は、『XFS for Linux Administration』を参照してマウントせずに xfs_repair を使用する方法を試してください。

1. 次の例では、root ファイルシステムが IX ブリックの先頭のディスク・ベイにあるディスクのパーティション 3 にあることを前提としています。システムからアクセス可能なパーティションの一覧を表示するには、次の例に示すコマンドを実行します。

   sh-2.05# cat /proc/partitions

```
major minor #blocks name rio rmerge rsect ruse wio wmerge wsect wuse running use aveq
 5     0    125470 xscsi/pci01.01.0/target0/lun0/disc 218 4524 18968 10952 0 0 0 0 0 10952 10952
 4     0  35843686 xscsi/pci01.03.0-1/target1/lun0/disc 32 101 692 69 2 14 512 10 0 80 80
 4     1    513008 xscsi/pci01.03.0-1/target1/lun0/part1 0 0 0 0 0 0 0 0 0 0 0
 4     2   9438208 xscsi/pci01.03.0-1/target1/lun0/part2 0 0 0 0 0 0 0 0 0 0 0
 4     3  25891840 xscsi/pci01.03.0-1/target1/lun0/part3 28 29 584 50 2 14 512 10 0 61 61
 4    16  35843686 xscsi/pci01.03.0-1/target2/lun0/disc 4 72 108 19 0 0 0 0 0 19 19
 4    17    513008 xscsi/pci01.03.0-1/target2/lun0/part1 0 0 0 0 0 0 0 0 0 0 0
 4    18   9438208 xscsi/pci01.03.0-1/target2/lun0/part2 0 0 0 0 0 0 0 0 0 0 0
 4    19  25891840 xscsi/pci01.03.0-1/target2/lun0/part3 0 0 0 0 0 0 0 0 0 0 0
```

2. システムが問題なく実行中である場合は、次の例に示すような XSCSI パス名が IX ブリックのベイ 1 にあるシステム・ディスクを表します。この例では、root ファイルシステムはパーティション 3 に置かれていると仮定しています。

   /dev/xscsi/pci01.03.0-1/target1/lun0/part3

3. ファイルシステムをマニュアルでマウントするには、次のコマンドを実行します。

```
mkdir /mnt/sysimage
mount -t xfs /dev/xscsi/pci01.03.0-1/target1/lun0/part3 /mnt/sysimage
```

サポートされる製品

この章では、SGI LX 3000 システムでサポートされる製品コンポーネントについて説明します(すべての製品の一覧は 18 ページの 表 3-1 を参照してください)。

この章では、各製品コンポーネントの説明を次のグループに分けて説明します。

- OS 拡張
- I/O サブシステム
- HPC アプリケーション・ツールとそのサポート
- システム管理ツール
- NUMA tools

SGI ProPack for Linux release で SGI から供給されるソフトウェアは、SGI LX 3000 用カーネル RPM、SGI Linux Environment 7.2 RPM セット、および SGI システム上での使用を前提とした

SGI ソフトウェアから構成されます。SGI ProPack for Linux 製品の一覧と説明を表 3-1 に示します。

表 3-1 SGI ProPack v2.1.1 for Linux 製品

製品	説明
Application performance measuring tools	以下のプログラム最適化ツールを含みます。 VTUNE - このツールは Intel によって開発、サポートされます。Itanium プロセッサのパフォーマンス測定機能を利用して、消費時間やその他のプロセッサ・アーキテクチャ・イベントをプロファイリングします。プロファイリング・データはアプリケーションのパフォーマンス解析、チューニング、改良に利用できます。 pfmon- このツールはオープン・ソースで、GPL ライセンスに沿って配布されます。Itanium プロセッサのパフォーマンス測定機能を制御するコマンド・ライン・インターフェイスを提供します。このツールが出力するデータを加工して、各種のアプリケーション・パフォーマンスのレポートが作成できます。レポートはアプリケーション・パフォーマンスの評価、チューニング、改良に利用できます。pfmon パッケージにはカスタム・パフォーマンス・ツールを作成するためのライブラリ・インターフェイス `libpfmon.a` も含まれます。
Array Services	SGI システム・クラスタで実行されるシステムと並列アプリケーションの管理を容易にするための、カーネル・サポートを備えたツールを提供します。
CpuMemSets	プロセッサとメモリの配置を実装するインフラストラクチャとカーネル・サポートを提供します。
cpuset	CPU セット指定機能を提供します。
CSA	ジョブ・ベースのアカウンティング機能を提供します。Linux システム上で、特定のログイン・アカウントに対してタスク単位のリソースおよびディスク消費のアカウンティングを行います。
FLEXlm	Floating License のためのランタイム環境です。Floating License を処理するためのデーモン群を含みます。
IOC4 driver	内臓 IDE CD-ROM、NVRAM、リアルタイム・クロック制御用ドライバです。

表 3-1 SGI ProPack v2.1.1 for Linux 製品 **（続き）**

製品	説明
kdb	実行中のシステムに対するカーネル・デバッグをサポートします。キーボードからの直接制御とシリアル・コンソールからの制御をサポートします。
Kernel partitioning support	パーティション・システムをサポートするために必要なソフトウェア・インフラストラクチャを提供します。パーティション間通信サポートを含みます。
Kernel performance improvements	コミュニティ・ベースで作成されたパッチを含みます。2.5.xカーネルおよびSGIカーネルでのO(1)スケジューラのサポート、Big Kernel Lock (BKL) でのロック競合を削減するためのパッチ、FRlocks (Fast-Reader Locks) による `xtime_lock`（`gettimeofday` が使用）の競合の削減のためのパッチなどが含まれます。
L1 and L2 System Controller firmware	ブリックとシステム・コンピュータ・ラック内の電源、冷却ファン、テスト機能を管理・モニタするためのサポートを提供します。
LKCD	システム・クラッシュ・ダンプ解析ツールを提供します。`lcrash` と、システム・クラッシュ・ダンプの保存と設定に必要な全ユーザ・レベル・スクリプトが含まれます。
MPT	SGI システム用に最適化された、業界標準のメッセージ交換ライブラリです。
NUMA Tools	NUMA 関連ツールのコレクション（`dlook`、`dplace` など）です。
Performance Co-Pilot collector infrastructure	大規模システム用のパフォーマンス・モニタとパフォーマンス解析サービスです。
PROM	システム・ブートとソフトウェア・インストール機能を提供します。
runon	コマンドを特定のCPUまたはCPU群で実行できます。
SGI Linux Environment 7.2 updates	SGI ProPack 同梱の SGI Linux Environment 7.2 ベースに対するアップデート RPM セットです。
XFS	Linux にハイパフォーマンス・ファイルシステムを導入します。

表 3-1 SGI ProPack v2.1.1 for Linux 製品 **（続き）**

製品	説明
XSCSI infrastructure	QLogic QLA12160 SCSI ホスト・バス・アダプタと ATAPI CDROM、ディスクをサポートします。QLogic Fibre Channel のホスト・バス・アダプタもサポートされます。サポートされるカードについては『SGI LX 3000 User's Guide』を参照してください。インフラストラクチャとして、エラー修正ハンドリング、フェイルオーバー、SAN のサポートを含みます。Linux SCSI レイヤと共に、あるいは単独で使用できます。Linux SCSI レイヤと協調動作するように設定すると、一般的なディスク・デバイス名（*/dev/sda* など）が使用されます。
XVM	ディスクのストライプ化、ミラーリングなどのソフトウェア・ボリューム管理機能を提供します。

次のソフトウェアは SGI によってサポートされません。

- SGI 以外から供給されるソフトウェア
- Red Hat の発表する他のリリース、アップデート、パッチ
- Linux コミュニティ、あるいは他のベンダーから入手したソフトウェア・パッチ、ドライバ、あるいは他のソフトウェア改変
- SGI の指定する以外のパラメータ設定やその他のモジュールを使って再コンパイルまたは再設定されたカーネル。特に、ext3、ReiserFS、LVM、md カーネル・コンポーネントは品質、機能、パフォーマンスに著しく影響するため、サポートされません。これらのコンポーネントの替わりに、XFS と XVM を使用してください。
- サポート外のハードウェア構成やデバイス

OS 拡張

SGI は、世界中で絶え間なく拡張・改善され続ける Linux OS の商用・企業環境の開発の中でも特に、大規模計算・大規模データに対するハイパフォーマンス・コンピューティング (HPC) 環境に重点を置いています。このために、SGI では IRIX OS と MIPS プロセッサ・ベースのシステムで培った NUMAflex と HPC の技術を活用し、Linux カーネルの中でも HPC 環境において重要な機能の改善に努めています。

CpuMemSets のサポート

CpuMemSets は、複数 CPU のスケジューリングと複数ノードのメモリ・アロケーションを行うシステム･サービスやアプリケーションに必要な Linux カーネル機能を提供します。SGI ProPack のカーネルとライブラリをインストールすることで、CpuMemSets サポートは有効になります。

デフォルト設定では、すべての CPU とメモリがすべてのアプリケーションで使用可能です。CpuMemSets 機能を利用して、任意のプロセス、プロセス・ファミリ、あるいはプロセス仮想メモリ領域を、システム中の特定の CPU 群やメモリ部分に制限されるように指定できます。

runon コマンドは CpuMemSets を利用して、特定のコマンドを特定の CPU 群の上で実行させます。CpuMemSets システム・インターフェイスにアクセスするために、C 共有ライブラリと Python 言語モジュールが供給されます。

SGI ProPack for Linux の将来のリリースでは、CpuMemSets 機能に簡単にアクセスできるその他のシステム・サービスが追加される計画です。

CpuMemSets サポートに関するドキュメント情報やその他の詳細は、『Resource Administration Guide for Linux』の「CPU Memory Sets and Scheduling」という題の章を参照してください。CpuMemSets RPM の一部としてインストールされるファイルの一覧は次のコマンドで参照できます。

```
rpm -ql CpuMemSets
```

また、*cpumemsets* と *runon* のマン・ページにも重要な情報が含まれています。

CpuMemSets は SGI のオープン・ソース・プロジェクトです。CpuMemSets は次のサイトからも入手可能です。

```
http://oss.sgi.com/projects/cpumemsets
```

Cpuset のサポート

Cpuset System は、あるプロセスまたはプロセス群が使用できるプロセッサの数を限定するための機能を提供するワークロード管理ツールです。

Cpuset は、特定の CPU の集合に名前を付けたもので、制限付き (restricted) または開放 (open) として定義されます。制限付き Cpuset は、その Cpuset に属するプロセスだけがその CPU 群で実行できます。開放 Cpuset では任意のプロセスが実行できますが、その Cpuset に属するプロセスはその CPU 群内でのみ実行されます。Cpuset は Cpuset 設定ファイルと名前によって定義されます。

大規模システムのシステム管理者は、Cpuset を使って CPU 区画を設定できます。各区画システムでは、特定のプロセス群を特定の CPU 群を使って実行でき、これらのプロセスとシステムの他の処理との間の干渉や競合を軽減します。制限付き Cpuset の場合、Cpuset にアタッチされたプロセスはシステム上の他の処理の影響を受けません。Cpuset にアタッチされたプロセスだけが、Cpuset 内の CPU 群で実行されます。開放 Cpuset は、特定のプロセスをある一定の CPU 群でのみ実行させて、システムの他の部分への影響を最小限に抑えるために使用できます。

Cpuset サポートのドキュメント情報と詳細については『Resource Administration Guide for Linux』の「Cpuset System」という題の章を参照してください。

完全システム・アカウンティング (CSA:Comprehensive System Accounting)

完全システム・アカウンティング (CSA: Comprehensive System Accounting) ソフトウェア・パッケージは SGI と Los Alamos National Laboratory (LANL) の共同オープン・ソースを IRIX から Linux に移植したもので、ジョブ・ベースのアカウンティング機能を提供します。Linux システム上で、特定のログイン・アカウント群に対してタスク単位のリソースおよびディスク消費のアカウンティングを行います。

高機能システム・アカウンティング機能は超大規模システム、特に複数の組織で共同利用されるシステムで重要になります。CSA はジョブ・コンテナ機能を使って Linux 上でジョブ概念を定

義します。ジョブはいくつかのプロセスを内包するコンテナで、任意のシステム・エントリ・ポイント（インタラクティブ・ログイン、cron ジョブ、リモート・ログイン、バッチ処理など）へのリソース配分を把握するために用いられます。

Linux版CSAはSGIのオープン・ソース・プロジェクトです。CSAは次のサイトから入手可能です。

```
http://oss.sgi.com/projects/csa
```

CSA サポートに関するドキュメント情報やその他の詳細は、『Resource Administration Guide for Linux』の「Comprehensive System Accounting」という題の章を参照してください。

システム・パーティション

SGI LX 3000 システムでは、大規模システムをより小さなシステム・パーティション群（クラスタ）に分割して、個々のパーティションでそれぞれ独立した OS カーネルを実行できます。パーティション設定としてたとえば 64 プロセッサ・8 パーティションの構成を取ることができ、また、より大規模にスケーリングすることも可能です。SGI パーティション設定ではグローバル共有メモリがサポートされます。あるパーティションで実行されているアプリケーションがグローバル共有メモリ・セグメントを作成し、その上のデータを他のパーティションで実行されているアプリケーションと共有することが可能です。

ハイパフォーマンス・コンピューティング (HPC) 問題を解くための大規模システムに対する要求の高まりに対して、大量の CPU、メモリ、I/O リソースのプールを管理するために OS 処理が他のワークロードに対するボトルネックとなる問題が顕在化しています。パーティションはこのような問題、特に MPI(Message Passing interface) ジョブのような並列ワークロードに適しています。各パーティション間の接続は（SGI のハイパフォーマンス NUMAlink 接続を使って）維持されているため、単一システム・イメージでは間に合わないような規模の計算が必要な場合でも、プロセス間の通信を低遅延・広帯域で処理できます。

パーティションの導入は、パフォーマンス改善以外にも HPC システムの可用性を高めるという利点を持ちます。大規模システムをより小さな複数のパーティションに分割し、パーティションの集合をクラスタ群として使用することで、単一のパーティション（= クラスタ・ノード）のハードウェアや OS のエラーによってシステム全体がダウンしたり、他のパーティションで実行されているソフトウェアが停止するのを回避できます。また、あるパーティションがハードウェア障害（たとえば C ブリック障害）によってダウンした場合、その不良 C ブリックはただちにシステムから切り離され、パーティションが再起動されて不良 C ブリックを除いてサービスを継続でき

ます。C ブリックの修理・交換が可能になった時点で、システムの実行中にブリックをウォーム・スワップできます。その後、都合の良いタイミングでシステムを再起動すれば、新しい C ブリックを含めてシステムを 100% 稼動させることができます。

I/O サブシステム

ある種の HPC ワークロードは主に CPU リソースを消費するのに対し、大量のデータを処理し、メモリと外部記憶の間でデータを高速転送するために、または大規模外部記憶システムを効率的に利用するために I/O サブシステムの能力を要求するワークロードもあります。IRIX から移植された SCSI サブシステム、XFS ファイルシステム、XVM ボリューム・マネージャや、その他のデータ転送機能は、安定性と頑健性に富むハイパフォーマンス・ストレージ I/O サブシステムを Linux に導入します。

ここでは、PCI-X バス固定ナンバリング、Ethernet デバイス固定命名、XSCSI サブシステム、XFS ファイルシステム、XVM ボリューム・マネージャの概要について説明します。

PCI-X バス固定ナンバリング

PCI-X バスの固定ナンバリングにより、システムの障害や再設定のために再起動を行ってもバス番号が維持されます。プラットフォーム初期化中にバスが検出されると、論理バス番号が割当てられます。それぞれのバスにはシステム全体で一意な論理番号が割当てられます。SGI LX 3000 システムでは、デフォルトで 256 個のバス（バス番号 0-255）がサポートされます。

デフォルトでは、I/O ブリックが接続されている最下位の C ブリック・モジュール ID からバス番号が割当てられていきます。I/O ブリックとは、SGI LX 3000 システムの PX ブリック、または IX ブリックです。各 I/O ブリックには 0x10 個のバスが割当てられますが、実際には、現時点では各 I/O ブリックは 6 個のバスしかサポートしません。このため、システム全体ではバス番号は連続せず、空隙が発生します。

ある I/O ブリックに 6 個のバスが物理的に存在すると仮定します。これらのバスには 0x1 から 0x6 までの番号が、I/O ブリック背面から見て左から右への順序で割当てられます。システム中に複数の I/O ブリックが存在する場合は、次の I/O ブリックのバスには 0x11 から 0x16 までの番号が割当てられます。この番号の最下桁は、各 I/O ブリックのバス位置に刻印されている 1 から始まる番号に対応します。

1 個の I/O ブリックしかない場合は、バスの固定ナンバリングは不要です。ただし、複数の I/O ブリックがシステム中に存在する場合は、固定ナンバリングの使用が強く推奨されます。I/O ブリックの 1 個にブート時障害が発生しても、バス番号は変更されません。

バス固定ナンバリングを使用するには、カーネルに対して ioconfig= パラメータを指定する必要があります。

ioconfig= パラメータはカンマで区切られた I/O ブリック番号のリストを引数に取ります。次の行は、elilo によるマニュアル・ブートの例を示しています。この例では、カーネルに対して I/O ブリック 101.01 にバス番号 0x1 から 0x6 までを、I/O ブリック 101.02 にバス番号 0x11 から 0x16 までを割当てるように指定しています。

```
Shell> elilo vmlinux ioconfig="101.01,101.02"root=/dev/xscsi/pci01.03.0-1/target1/lun0/part3
```

ioconfig で指定する数字は、L2 コントローラからシステムを調べることによってわかります。例を次に示します。

```
l2-pumpkin-001-L2>pwr
001c11:
power appears on
001c27:
power appears on
101i01:
power appears on
101p02:
power appears on
```

上の例で、101i01 は前述のブート・オプションで示した IX ブリック 101.01 を表しています。同様に 101p02 は PX ブリック 101.02 を表します。

多くの場合、システムを自動ブートできるように設定します。このためには、elilo.conf ファイルで ioconfig パラメータを設定します。実行中の Linux システムでの elilo.conf ファイルのパスは /boot/efi/elilo.conf です。elilo.conf ファイルの例を次に示します。

```
prompt
timeout=50
relocatable
default=sgilinux
append="ioconfig=101.01,101.02"

image=vmlinuz-2.4.19-sgi21r4
        label=sgilinux
        read-only
        root=/dev/xscsi/pci01.03.0-1/target1/lun0/part3
```

上の例で、append= 行に注意してください。elilo.conf ファイルでは、ここに ioconfig 情報を記述します。

Ethernet デバイスの固定命名

Ethernet デバイスに対する固定命名は SGI 独自のメカニズムで、SGI LX 3000 システムにも採用されています。固定命名とは、SGI LX 3000 システムの IO9 インターフェイスに接続された Gigabit Ethernet カードを常に eth0 に設定するためのメカニズムです。このメカニズムにより、SGI LX 3000 システムに複数の Ethernet デバイスが存在する場合でも、MAC アドレスを基準にしてベース Ethernet デバイス番号を正しく割当てます。

/etc/sysconfig/networking/eth0_persist ファイルに Ethernet デバイス番号と MAC アドレスの対応が記述されます。このファイルが存在しないとシステムのブート時に /etc/rc.d/init.d/eth_persist スクリプトが実行されて、ファイルが生成されます。実際に IO9 Ethernet カードの MAC アドレスに eth0 を確実に割当てるには、SGI LX 3000 システムを最初に起動した後に、このファイルの編集が必要な場合があります。

固定命名によって、Ethernet カードを追加しても Ethernet デバイス番号と MAC アドレスの対応が維持されることを保証する以外に、Ethernet デバイスの番号付けを制御できます。たとえばデバイス番号 ethX の Ethernet カードが欠けたとき、その機能を Ethernet カード ethY で補いたい場合は */etc/sysconfig/networking/eth0_persist* ファイルを編集することでその Ethernet カードに番号 ethX を振り直すことができます。

/etc/sysconfig/networking/eth0_persist ファイルの例を次に示します。

```
eth0 08:00:69:13:dc:ec
eth1 08:00:69:13:72:e8
```

このファイルにより、システムは次のように設定されます。

```
[root]# ifconfig -a
eth0 Link encap:Ethernet  HWaddr 08:00:69:13:DC:EC
     inet addr:128.162.246.125 Bcast:128.162.246.255 Mask:255.255.255.0
     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:843 errors:0 dropped:0 overruns:0 frame:0
     TX packets:1245 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:386044 (376.9 Kb)  TX bytes:126741 (123.7 Kb)
     Interrupt:59
```

```
eth1 Link encap:Ethernet  HWaddr 08:00:69:13:72:E8
     BROADCAST MULTICAST  MTU:1500  Metric:1
     RX packets:136 errors:0 dropped:0 overruns:0 frame:0
     TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:8850 (8.6 Kb)  TX bytes:0 (0.0 b)
     Interrupt:63
```

IO9 のすぐ右のスロット（バス 1: スロット 2）に Ethernet ボードを接続して、ボード接続後に OS をインストールした場合または固定命名ファイル `eth0_persist` を削除した場合は、IO9 は `eth0` ではなく `eth1` になります。この設定は製品ライセンスに影響する場合があるので、推奨されません。Ethernet ボードを他の IX ブリック・スロットに接続すれば、この問題は発生しません。

バス 1 のスロット 2 に Ethernet ボードを接続する必要がある場合は、Ethernet ボードを接続しない状態でまず ProPack を含む OS をインストールします。ProPack インストール後、Ethernet ボードをバス 1 のスロット 2 に取付けます。

XSCSI サブシステム

IRIX システムでの商用実績と品質を備えた SGI XSCSI システムを Linux に付加することで、頑健なエラー・ハンドリングとフェイルオーバー機能、ストレージ・エリア・ネットワーク (SAN) インフラストラクチャがサポートされます。XSCSI は、IRIX システムで長年にわたって大規模システム向けにパフォーマンス・チューニングされています。XSCSI は、標準的に使われているオープン・ソース・ドライバそのままでは対応できないような SGI 独自アーキテクチャの機能を活用します。ここでは、XSCSI のデバイス命名法について説明します。

XSCSI デバイス命名法

XSCSI デバイスには次のような名前が付けられます。

```
/dev/xscsi/pci01.03.0-1/target1/lun0/part1
```

XSCSI デバイス名は次のコンポーネントから構成されます。

`pci01`	システム・バス番号 0x1
`03`	バス内でのデバイス番号 3
`0-1`	論理ユニット 0、ポート 1

デバイス番号（スロット番号）、論理ユニット、ポート番号は固定です。これらの値が変更されることはありません。ただし、システム・バス番号はハードウェア障害（I/O ブリックがブートしない等）や再設定によって変更する可能性があります。

PCI-X のバス固定ナンバリング（24 ページの「PCI-X バス固定ナンバリング」参照）が有効になっている場合はバス番号が固定され、ハードウェアの障害や再構成があってもバス番号が変更されることはありません。XSCSI 名をマウントやデバイス指定に使用するときは、バス固定ナンバリングを使用すると、その XSCSI デバイス名は再起動を経ても変更されません。

XFS ファイルシステム

SGI XFS ファイルシステムは、業界トップレベルのハイパフォーマンス・ファイルシステムを Linux に導入します。XFS はオープン・ソースの、高速復帰性を持つジャーナル・ファイルシステムで、ダイレクト I/O サポート、スペース・プリアロケーション、アクセス・コントロール・リスト、quotas、その他の商用ファイルシステム機能を備えています。Linux 上の他のファイルシステムも使用可能ですが、IRIX で長年にわたってパフォーマンス・チューニングと改善を重ねてきた XFS は、HPC 環境で必要になる大規模なデータ I/O ワークロードに特に適しています。

XFS ファイルシステムの詳細は『Linux XFS Filesystem Administration』を参照してください。

XVM ボリューム・マネージャ

SGI XVM ボリューム・マネージャは、複数の物理ディスク記憶装置を 1 個の論理的な単位（論理ボリューム）として結合するための論理構造を提供します。論理ボリュームは標準的なパーティションと同じように扱えるため、パーティションが使えるどのような引数にも指定できます。

論理ボリュームを使用することで、単一の物理ディスクよりも大きな raw デバイスやファイルシステムを構成できます。論理ボリュームは複数のディスクをストライプ化することも可能なので、ディスク I/O パフォーマンスの改善にも利用できます。また、他のディスク上のデータのミラー化にも利用できます。

HPC アプリケーションのツールとサポート

SGI IRIX から、HPC ライブラリ、ツール、およびソフトウェア・パッケージが Linux と Itanium 2 をベースにした高性能 HPC 環境のためのシステムへ移植されています。

Message Passing Toolkit

SGI Message Passing Toolkit (MPT) は、SGI システム用に最適化された業界標準メッセージ交換ライブラリを提供します。Linux では、MPT は MPI と SHMEM API を含み、メモリ間転送を高速に行うブロック転送エンジン (BTE) などの SGI ハードウェアを低階層で使用できるようにして、ハードウェア・メモリ・コントローラのフェッチ (fetchop) をサポートします。fetchop は複数の MPT プロセス間の直接通信と同期をサポートし、OS システムコールの使用によるオーバーヘッドを避けます。

MPI ジョブのような並列ワークロードは、SGI Array Services ソフトウェアを使用することでクラスタまたはパーティション・システム上で起動、制御、監視できます。Array Services によって、クラスタの複数ノードや複数のシステム・パーティションで実行できるプロセスのセットを定義するための array セッションを記述できます。Array Services はプロセス・コンテナを提供するカーネル・モジュール Process Aggregates (PAGGs) を使って実装されています。PAGGs は SGI によって Linux 上でオープン・ソース化されています。

Message Passing Toolkit の詳細については『Message Passing Toolkit: MPI Programmer's Manual』を参照してください。

Performance Co-Pilot (PCP)

SGI PCP ソフトウェアは大規模システム用に設計されたパフォーマンス・モニタとパフォーマンス管理のサービスのセットで、IRIX から Linux へ移植されています。PCP は低階層のハードウェア・カウンタや MPT と統合され、CPU、I/O、ネットワークの統計サービス、ビジュアル化ツール、モニタ・ツールなどを提供します。

システム管理

ここではシステム管理ツールについて説明します。システム管理ツールにはシステム起動に必要なハードウェア・ソフトウェア環境、ライセンス管理、システム・コンソール、およびシステム・コントローラが含まれます。

PROM チップ

PROM (Programmable read only memory) チップはシステム・ハードウェアに組み込まれているメモリで、CPU のブート、システム管理、ソフトウェアのインストールなどのために使用されます。PROM チップはディスクや OS の一部ではなく、システムの最下層に配置されます。PROM は内容を消去したり、実行をバイパスできません。PROM の詳細は `prom`(1) マン・ページを参照してください。

拡張ファームウェア・インターフェイス (EFI: Extensible Firmware Interface)

SGI LX 3000 には、CPU への入出力をサポートするために EFI プラットフォームが用意されています。また EFI は、サーバのブート設定を管理し、ブート・メニューを不揮発性メモリに保存します。

EFI プロンプトからは、基本的なファイル管理（テキスト編集を含む）、設定の変更、ブート時に実行されるスクリプトの作成などのタスクが実行できます。EFI コマンドの概要を表 3-2 に示します。

表 3-2 EFI コマンド

EFI コマンド	説明
`alias` [`-bdv`] [*sname*] [*value*]	エイリアスを設定、表示します。
`attrib` [`-b`] [+/- *rhs*] [*file*]	ファイル属性を設定、表示します。
`bcfg`	ブート・ドライバの設定とオプションのロードを行います。
`cd` [*path*]	カレント・ディレクトリを変更します。
`cls` [*background color*]	画面をクリアします。

表 3-2 EFI コマンド **（続き）**

EFI コマンド	説明
`comp` *file1 file2*	2 つのファイルを比較します。
`cp` *file* [*file*] ... [*dest*]	ファイルやディレクトリをコピーします。
`date` [*mm/dd/yyyy*]	日付を設定、表示します。
`dblk` *device* [*Lba*] [*blocks*]	ブロック I/O デバイスの 16 進ダンプを出力します。
`dh` [`-b`] [`-p` *prot_id*]	[*handle*]	ハンドル情報を出力します。
`dmpstore`	変数ストアをダンプします。
`echo` [`-on`	`-off`]	[*text*]	テキストを標準出力にエコー出力したり、スクリプトのエコー出力のオン・オフを制御します。
`edit` [*file name*]	ファイルを編集します。
`endfor`	（スクリプト内のみ）ループの末尾を示します。
`endif`	（スクリプト内のみ）`IF THEN` 構文の末尾を示します。
`err` [*level*]	エラー・レベルを設定、表示します。
`exit`	終了します。
`for var in` *set*	（スクリプト内のみ）ループ構文を始めます。
`getmtc`	単調カウンタ値を取得します。
`goto` *label*	（スクリプト内のみ）ラベル位置にジャンプします。
`guid` [`-b`] [*sname*]	既知 `guid` ID 一覧を表示します。
`help` [`-b`] [*internal command*]	コマンド・ヘルプを表示します。
`if` [`not`] *condition* `then`	（スクリプト内のみ）`IF THEN` 構文を始めます。
`load` *driver_name*	ドライバをロードします。
`ls` [`-b`] [*dir*] [*dir*] ...	ディレクトリの内容を表示します。
`map` [`-bdvr`] [*sname*[:]] [*handle*]	デバイス・パスに対する短縮名をマッピングします。
`mem` [*address*] [*size*] [*;MMIO*]	メモリやメモリ・マップド I/O をダンプします。
`memmap` [`-b`]	メモリ・マップをダンプします。
`mkdir` *dir* [*dir*] ...	ディレクトリを作成します。
`mm` *address* [*width*] [*;type*]	Mem、MMIO、IO、PCI のメモリ設定を変更します。

表 3-2 EFI コマンド **（続き）**

EFI コマンド	説明
`mode` [*col row*]	テキスト・モードを設定、表示します。
`mount` *BlkDevice* [*sname*[`:`]]	ブロック・デバイス上のファイルシステムをマウントします。
`mv` *sfile dfile*	ファイルを移動します。
`pause`	（スクリプト内のみ）終了 / 継続を尋ねるプロンプトを出力します。
`pci` [*bus dev*] [*func*]	PCI デバイス情報を表示します。
`reset` [`cold`/warm] [*reset string*]	リセットの設定 (Cold/Warm) を示します。
`rm` *file*/*dir* [*file*/*dir*]	ファイルやディレクトリを削除します。
`set` [`-bdv`] [*sname*] [*value*]	環境変数を設定、表示します。
`setsize` *newsize fname*	ファイルのサイズを設定します。
`stall` *microseconds*	*x* ms 間の遅延を作ります。
`time` [*hh*:*mm*:*ss*]	時刻を設定、表示します。
`touch` [*filename*]	ファイル属性を設定、表示します。
`type` [`-a`] [`-u`] [`-b`] *file*	ファイル内容を出力します。
`ver`	バージョン情報を表示します。
`vol` *fs* [*volume label*]	ボリューム・ラベルを設定、表示します。

FLEXlm

FLEXlm は Globetrotter Software から提供される、頑健性と柔軟性を備えたライセンス管理システムです。サードパーティは開発する製品に簡単にライセンス機能を追加でき、システム管理者は最低限の労力でライセンスのインストールと管理が行えます。FLEXlm では、単純なノードロック・ライセンスと冗長サーバで有効なフローティング・ライセンス、そしてそれらに伴う幅広いライセンス・オプションがサポートされます。

ライセンス付きのソフトウェアをビルドするには、開発ベンダーは Globetrotter Software からキーのセットを購入する必要があります。システム管理者は任意の場所にライセンス・サーバを

インストールできます。SGI が販売する製品は通常、FLEXlm を使ってライセンスが設定されています。

詳細については http://www.globetrotter.com/flexlm/flexlm.shtml を参照してください。

SGIconsole

SGIconsole は、IRIX OS および SGI ProPack for Linux を実行する SGI サーバ群をコンソールから管理、モニタするためのハードウェアとソフトウェアです。サーバには SGI LX 3000 サーバなどの、SGI パーティション・システムや 単一システム・イメージ・サーバなどを含められます。

SGIconsole は 1U ラックマウント・タイプの Intel Pentium ベース SGI サーバ、シリアル・マルチプレクサまたは Ethernet ハブ、および Console Manager パッケージと Performance Co-Pilot（他のハードウェア、ソフトウェアのリモート管理ツールにアクセスします）を含むソフトウェア・パッケージから構成されます。

Console Manager は SGIconsole ツールの GUI となり、複数の SGI サーバを制御するために使用されます。SGIconsole はコマンドライン・インターフェイスからも使用できます。SGIconsole の詳細については『SGIconsole 1.1 Start Here』を参照してください。

システム・コントローラ・ファームウェア

L1/L2 コントローラは SGI システムのシステム・コントローラ・ファームウェアです。

SGI LX 3000 システム と SGI Origin / Onyx 3000 Series システムの各ブリックには L1 コントローラが内蔵されています。L1 コントローラは各ブリックの電源投入と制御のシーケンス、温度と電圧のモニタを行います。

L2 コントローラはラック・レベルのコントローラで、ラック内のブリックを監視、制御します。システム内の L2 コントローラ同士はネットワーク接続されて、各ブリックの制御と監視の情報を交換し、システム全体の制御と監視に利用します。

L1/L2 システム・コントローラ・ファームウェアについての詳細は『SGI L1 and L2 Controller Software User's Guide』を参照してください。

NUMA Tools

NUMA Tools は NUMA 関連のツール・コレクションです。現在、次のコマンドが供給されています。

dlook コマンド

dlook コマンドは指定したプロセスのメモリ・マップと CPU 利用状況を表示します。プロセス仮想アドレス空間の各ページについて、次の情報が出力されます。

- ページを所有するオブジェクト（ファイル、SYSV 共有メモリ、デバイス・ドライバなど）
- ページ・タイプ（RAM、FETCHOP、IOSPACE など）
- RAM メモリでは次の情報が出力されます。
  - メモリ属性（SHARED、DIRTY など）
  - ページが配置されているノード
  - ページ物理アドレス（オプション）

オプションとして、プロセスが使用した各システム物理 CPU の 消費 CPU time も表示できます。

dplace コマンド

dplace コマンドは関連するプロセス群を特定の CPU（ノード）のグループにバインドして、プロセスの移行が発生するのを防ぎます。このツールはメモリ・アクセスをローカル・ノードに集中させることにより、ある種のソフトウェアのパフォーマンスを改善できます。

パフォーマンス・チューニング

この章では、シングルプロセッサまたはマルチプロセッサによるプログラミングでのチューニングに関する話題を取り上げます。本章の内容は次の各節に分かれます。

- シングルプロセッサ・プログラムのパフォーマンス・チューニングについては 36 ページの「シングルプロセッサ・コードのチューニング」を参照してください。
- マルチプロセッサ・プログラムのパフォーマンス・チューニングについては 45 ページの「マルチプロセッサ・コードのチューニング」を参照してください。
- シングルプロセッサ、マルチプロセッサの双方のプログラミングのパフォーマンス・チューニングで利用できるツールについては 50 ページの「プロファイラとパフォーマンス・ツール」を参照してください。

xiv ページの「関連ドキュメント」に挙げた参考文献も参照してください。

システム構成の確認

使用中のシステムの詳しい状態は *`/proc`* 疑似ファイルシステムで調べられます(詳細は `proc`(5) マン・ページを参照してください)。たとえば、次のような情報が得られます。

- *`/proc/cpuinfo`* は各プロセッサの状態を出力します。クロック・スピードとプロセッサのステッピング番号がわかります。
- *`/proc/meminfo`* はシステム・メモリの全体的な使用状況（総容量、空き容量、スワップ容量など）を表示します。
- *`/proc/discontig`* はメモリ利用をページ単位で表示します。
- *`/proc/pal/cpu0/cache_info`* は L1/L2/L3 キャッシュ構造についての詳細（サイズ、レイテンシ、連想度、ライン・サイズなど）を表示します。*`/proc/pal/cpu0`* 内のその他のファイルは TLB (Translation Lookaside Buffer) 構造、クロック比、その他の詳細を表します。

- */proc/version* は現在インストールされているカーネルについての情報を表示します。
- */proc/perfmon* ファイルが */proc* に存在しない（エクスポートされていない）場合は、カーネルはパフォーマンス・カウンタを開始しておらず、カウンタを使用するパフォーマンス・ツールは動作しません。
- */proc/mounts* は現在マウントされているファイルシステムの詳細を表示します。
- */proc/modules* は現在インストールされているカーネル・モジュールについての詳細を表示します。

uname コマンドはカーネルのバージョンその他のシステム情報を出力します。また、*topology* コマンドは */dev/hw* の情報を元にシステム構成を表示する Bourne シェル・スクリプトです。詳細は *topology(1)* マン・ページを参照してください。

SGI LX 3000 システムで現在推奨されるコンパイラは efc および ecc（Intel Fortran コンパイラおよび Intel C/C++ コンパイラ）です。他のコンパイラ（GNU コンパイラなど）も使用できますが、efc と ecc が Linux システム上での最善のパフォーマンスを導きます。この章の説明では、特に注意がない限り efc / ecc コンパイラを前提とします。

シングルプロセッサ・コードのチューニング

シングルプロセッサ・コードのパフォーマンスをチューニングするときは、次の基本ステップに沿ってください。

- 正しい結果が得られていることを確認してからチューニングを開始します。詳細は 37 ページの「出力の正確性の確認」を参照してください。
- 数値演算ライブラリ、科学計算ライブラリなどのパッケージに含まれるチューニング済みの既存コードを使用します。詳細は 40 ページの「チューニング済みコードの使用」を参照してください。
- チューニングが必要な部分を決定します。詳細は 41 ページの「チューニング必要部分の決定」を参照してください。
- コンパイラを適切に実行します。詳細は 42 ページの「コンパイラ・オプションの選択と適用」を参照してください。

- キャッシュ・パフォーマンスのチューニングの可能性を検討します。詳細は 43 ページの「キャッシュ・パフォーマンスのチューニング」を参照してください。
- メモリ管理モードを変えるための環境変数を設定します。詳細は 43 ページの「メモリ管理」を参照してください。
- スタックとデータ・セグメントを変更します。詳細は 44 ページの「chatr によるスタックとデータ・セグメントの変更」を参照してください。

出力の正確性の確認

チューニングの最初にまず行うのは、正しい出力が得られていることを確認することです。出力の正確性が確認できたら、チューニングを開始できます。まず特別な最適化はすべて無効にし、デフォルトの最適化に絞って出力をチェックします。これは、コンパイラ・オプションとデバッグ・ツールを使って行います。

次のコンパイラ・オプションはパフォーマンスのトレースと移植に関して役立ちます。

- -O: -O0 はすべての最適化をオフにします。デフォルトは -O2 です。
- -g: -g オプションはデバッグ・シンボルを有効にします。
- -mp: -mp オプションは浮動小数点最適化を制限して、宣言された精度を守ります。
- -IPF_fltacc: -IPF_fltacc オプションは精度に影響する浮動小数点関連の最適化を無効にします。
- -r:、-i: -r8 および -i8 オプションを指定するとデフォルトの実数、整数、および論理値のサイズが 8 バイトになります。Cray コードを移植するときに効果があります（このオプションは intrinsic/external ライブラリ関数を明示的に宣言します）。

デバッグ・ツールは、出力の正確性を確認するために使用できます。次のデバッガが使用できます。

- gdb: GNU プロジェクトのデバッガです。C、C++、Fortran 95 で書かれたプログラムのデバッグに使用できます。C や C++ でコンパイルするときは、コンパイラ・コマンド・ラインに -g オプションを指定して gdb の使用する dwarf2 シンボル・データベースが生成されるようにします。

Fortranのデバッグにgdbを使用するときは-gおよび-O0オプションを指定します。Fortranプログラムで -O1 またはそれ以上のオプションを指定する場合は、gdb は使用しないで下さい。

Fortran 95 コード用のデバッガは *http://sourceforge.net/project/showfiles.php?group_id=56720* からダウンロードできます（標準の gdb コンパイラは Fortran 95 コードには対応していません）。正しいバージョンの gdb を使用しているかどうか確認するには gdb -v を実行します。出力が次のようになることを確認してください。

```
GNU gdb 5.1.1 FORTRAN95-20020628 (RC1)
Copyright 2002 Free Software Foundation, Inc.
```

gdb コマンドは *http://sources.redhat.com/gdb/onlinedocs/gdb_toc.html* からダウンロードできるユーザーズガイド、または gdb コマンドの help オプションで一覧が参照できます。現時点のバージョンの gdb は ar.ec レジスタの状態を正しく報告しません。アセンブリ・コード・レベルで、ソフトウェア・パイプラインを使用した、ローテーションする、レジスタ・ベースのループをデバッグする場合は、idb を使用してください。

- idb:Linux プラットフォーム用の完全な機能を備えた Intel 製のシンボリック・デバッガです。このデバッガは C、C++、FORTRAN 77、Fortran 90 で書かれたプログラムのデバッグを幅広くサポートします。現時点では、idb は Itanium システムで動作するマルチスレッド・プログラムやマルチプロセッサ・プログラムのデバッグには使用できません。

  シェル・コマンドで idb に -gdb オプションを付けて実行すると、使用できるユーザ・コマンドとデバッガ出力が gdb 風になります。

- ddd: コマンドライン・デバッガに接続する GUI です。gdb と idb がサポートされます。詳細は 38 ページの「ddd の使用」を参照してください。

ddd の使用

ddd は任意のコマンドライン・デバッガに接続できる GUI ツールです。デフォルトでは gdb に接続されます。新たに ddd を起動するときは --debugger オプションでデバッガが指定できます（例 : --debugger "idb"）。

起動後、画面の各ペインに次の情報が表示されます。

- 配列表示
- ソースコード
- 逆アセンブル・コード

- デバッガ・エンジンへのコマンドを入力するウィンドウ

各パネルのオン・オフは **[View]** メニューから制御できます。

頻繁に使うコマンドはメニューから実行できます。また、次の操作も覚えておくと便利です。

- アセンブラ画面でアドレスを選択してマウス右ボタンをクリックして **[lookup]** を選択すると、コマンド画面で `gdb` lookup コマンドが実行されて対応するソース行が表示されます。
- ソース画面で変数を選択してマウス右ボタンをクリックすると、現在の値が表示されます。配列は配列表示ウィンドウに表示されます。配列は **[Menu]**> **[Print Graph]** オプションを選択すると PostScript 形式で出力されます。
- レジスタ・ファイルの内容（general、floating-point、NaT、predicate、application）は **[Status]** メニューの **[Regsiters]** コマンドを選択すると表示されます。**[Status]** メニューからは、スタック・トレースの表示やスレッドの切り替えも行えます。

ヒープ破壊問題対策

`glibc` の `malloc/free` を使用してメモリを動的に管理するプログラムで発生する可能性のあるヒープ破壊問題には、2 つのチェック方法があります。1 つは環境変数を使用する方法、もう一つは Electronic Fence を使う方法です。

`MALLOC_CHECK_` 環境変数の値を 1 にすると診断メッセージが表示されます。2 にすると、ヒープ破壊が検出された瞬間に実行が中断されます。

ElectricFence は Linux および Unix 用の `malloc` デバッガです。このデバッガを使用すると、`malloc`() メモリ・バッファがオーバーラン（またはアンダーラン）した瞬間にプログラムを停止して、ソースコード中のバグ発生個所を表示します。このデバッガはメモリ割当て用ブロックの先頭または末尾を不正なページに隣接させて、エラー発生時のバッファオーバーランまたはアンダーランに対してセグメンテーション・エラーを起こします。また、すでに開放されたメモリ領域がアクセスされたときも検出されます。

アンダーラン / オーバーランは同時に検出できません。デフォルトの動作では、割当てられたメモリの直後にアクセス不能なページを配置しますが、環境変数 `EF_PROTECT_BELOW` を設定することで逆の状況も設定できます。Electric Fence を使用するには `libefence` ライブラリをリンクします。次の例を参照してください。

```
% cat foo.c
```

```
#include <stdio.h>
#include <stdlib.h>
int main (void)
{
    int i;
    int * a;
    float *b;

    a = (int *)malloc(1000*sizeof (int));
    b = (float *)malloc(1000*sizeof (float));

    a[0]=1;
    for (i=1 ; i<1001;i++)
        {
          a[i]=a[i-1]+1;
        }
    for (i=1 ; i<1000;i++)
        {
          b[i]=a[i-1]*3.14;
        }

    printf("answer is %d %f \n"a[999],b[999]);

}
```

このプログラムのコンパイル、実行は次のようになります（ライブラリを指定してコンパイルしたときのエラー表示に注意してください）。

```
% ecc foo.c
% ./a.out
answer is 1000 3136.860107
% ecc foo.c -lefence
% ./a.out

Electric Fence 2.2.0 Copyright (C) 1987-1999 Bruce Perens
Segmentation fault
%
```

巨大コア・ファイルの生成を回避するために推奨される方法は、デバッガから Electric Fence を使用することです。詳細は `efence` マン・ページを参照してください。

チューニング済みコードの使用

可能な場合は、ハードウェア・パフォーマンスに対して最適化されたチューニング済みコードを使用してください。

次の数値演算関数ライブラリを必要なときに使用することで、最善の結果が得られます。

- MKL: Intel の Math Kernel ライブラリです。BLAS、LAPACK、および FFT ルーチン群が含まれます。
- VML: Vector Math ライブラリは、MKL パッケージの一部 (`libmkl_vml_itp.so`) として提供されます。
- Standard Math library: 標準数値計算ライブラリ関数は Intel コンパイラの `libimf.a` で提供されます。`-lm` オプションが指定されると、`glibc` の `libm` ルーチンが優先的にリンクされます。

MKL と VML のドキュメンテーションは次の Web サイトで入手できます。

http://intel.com/software/products/perflib/index.htm?iid=ipp_home+software_libraries&

チューニング必要部分の決定

チューニングでは、コードの中で改善効率の高い部分を特定することが重要です。次のツールを活用してください。

`time`	ユーザ・タイム、システム・タイム、および経過時間を測定します。
`gprof`	プログラムの実行プロファイル（`pcsamp` プロファイル）が得られます。`-p` コンパイラ・オプションを指定すると `gprof` が使用可能になります。
VTune	Intel のパフォーマンス・モニタ・ツールで、Linux サーバ、Windows クライアントのアプリケーションです。複数の Itanium/Linux システムのリモート・サンプリングをサポートします。
`pfmon`	Itanium と Linux のために設計されたパフォーマンス・モニタ・ツールです。Itanium の Performance Monitoring Unit (PMU) を使ってバイナリを加工せずに直接カウント、サンプリングします。

VTune と `pfmon` の詳細は、50 ページの「プロファイラとパフォーマンス・ツール」を参照してください。

コンパイラ・オプションの選択と適用

コンパイラ・オプションには、パフォーマンス最適化に使用できるものがあります。efc や ecc のオプションの概要を見るには、-help オプションを付けてコンパイラをコマンドライン実行してください。-dryrun オプションを指定すると、efc や ecc が生成するドライバ・ツール・コマンドが表示されます。このオプションは実際にツールを実行しません。

次のオプションは、パフォーマンス・チューニングに利用できます。

-ftz　アンダーフローをゼロにしてカーネル・トラップを回避します。-O3 最適化レベルではデフォルトで有効になります。

-fno-alias　ポインタ・エイリアスがないものと見なします。エイリアス関連オプションにはこの他に -ansi_alias と -fno_fnalias があります。エイリアスを使用すると不正なコードが生成される可能性があることに注意してください。

-ip　単一ファイル内でのプロシージャ間の最適化を行います。

-ipo　複数ファイル間のプロシージャ間の最適化を行います。

-O3　-O2 レベルの最適化に加えて、ループ変換やプリフェッチなど、より積極的な最適化を行います。レベル 3 最適化は、必ずしもすべてのプログラムでパフォーマンス改善に寄与するとは限りません。

-opt_report　最適化に関するレポートを生成し、-opt_report_file で指定されるファイルに保存します。

-override_limits
公式にドキュメント化されているオプションではありませんが、コンパイラの内部制限を越えて最適化処理を継続することを許します。

-prof_gen / -prof_use
プロファイリング情報を生成、利用します。これらのオプションを使用するときは、次の 3 ステップの手順が必要です。

1. -prof_gen を指定してコンパイルします。
2. 試験データを使ってプログラムを実行します。
3. -prof_use を指定して、試験実行のデータを元にコンパイルします。

-S　コンパイルを実行して、アセンブル・リストを .s ファイルに生成しますが、リンクは行いません。アセンブル・リストは -opt_report オプション出力と照合して、ループ最適化の成否をチェックするために使用します。

キャッシュ・パフォーマンスのチューニング

キャッシュのパフォーマンスをチューニングするには、次のような方法があります。

- キャッシュのスラッシングを引き起こすような、大きな 2 の指数幅のストライドや次元指定は避けます。
- 可能な限りストライド 1 を使用します。
- 同じ 16 バイト幅バンクへのアクセスが 2 つ同時に発生すると、キャッシュ・バンクのコンフリクトが発生する可能性があります。`pfmon -e L2_OZQ_CANCELS1_BANK_CONF` コマンドの出力と `pfmon -e CPU_CYCLES` コマンドの出力に、トータル CPU サイクル数に比べて多数のバンク・コンフリクトが検出される場合は、配列パディングの方法を変えてみてください。2 つのコマンドは、次の行で 1 つにまとめることができます。

  ```
  % pfmon -e CPU_CYCLES,L2_OZQ_CANCELS1_BANK_CONF a.out
  ```

  パフォーマンスのためのイベント監視では、最大で同時に 4 つのイベントがカウントできます。
- 同時に使用するデータはグループ化し、コード中では可能な限りベクトルを使わないようにします。
- テンポラリ配列の使用は避け、データ・コピーをできるだけ減らします。

メモリ管理

`glibc` の `malloc/free` でメモリの割当と開放を頻繁に行うコードは、メモリ管理オーバーヘッドによってシステム時間を消費する傾向があります。デフォルトでは、`glibc` はシステム全体のメモリ効率をパフォーマンスよりも優先します。この話題の詳細は `http://www.linuxshowcase.org/ezolt.html` で解説されています。

ハイパフォーマンス指向のメモリ管理モードを有効にするには、次の環境変数を設定してください。

```
% setenv MALLOC_TRIM_THRESHOLD_ -1
% setenv MALLOC_MMAP_MAX_ 0
```

`efc` では組込みのメモリ割当ても内部的に `glibc malloc` を使用するので、これらの環境変数は `malloc/free` による Cray ポインタ等を使った Fortran コードでも原則的に有効です。ただし

Fortran 90 のアロケータブル配列は Fortran ライブラリ呼出しで直接管理されているため、効果はありません。

chatr によるスタックとデータ・セグメントの変更

chatr コマンドは、プログラムの ELF ヘッダを変更して、実行時にスタック・セグメントとデータ・セグメントにある命令が実行可能かどうかを切り替えます。

デフォルトでは、スタック・セグメントとデータ・セグメントの両方が実行可能です。特定のワークロードでは、デフォルトの設定をオフにしてスタックとコード・セグメントを実行不可にすることでパフォーマンスを改善できます。この手法は、多数の fork() 呼出しを伴うワークロードで特に効果があります。fork() 呼出しをあまり含まないワークロードでは、パフォーマンスの大幅な改善は見込めません。

sysctl コマンドを使って、この動作をシステム全体で変えることができます。次のコマンドは、スタックおよびデータ・セグメントを常に実行不可にします。

```
% sysctl vm.executable_stacks=0
```

次のコマンドは、スタックおよびデータ・セグメントを実行可能にすることを許可します。

```
% sysctl vm.executable_stacks=1
```

この変更で、プログラムによっては従来問題なかったものが SEGV でエラーを起こす可能性があります。問題を起こすのは通常、スタックまたはデータ・セグメントに命令を保存していて、そこへ分岐するプログラムです（たとえば古いバージョンの X サーバには、グラフィックス・ドライバをデータ・セグメントへロードするものがあります。Java JIT コンパイラも同様の問題を持ちます）。このようなプログラムも、chatr コマンドによって vm.executable_stacks の値に関係なく正しく実行できるようになります。

詳しい使用法については、chatr(1) マン・ページを参照してください。

マルチプロセッサ・コードのチューニング

マルチプロセッサ・コードのチューニングを行う前に、まずシングルプロセッサのチューニングを行ってください。多くの場合、マルチプロセッサ・コードでも良い結果が得られます。詳細は36 ページの「シングルプロセッサ・コードのチューニング」を参照してください。

マルチプロセッサ・チューニングは大別して次のステップに分かれます。

- コードの並列化手法を選択します。詳細は 45 ページの「コードの並列化」を参照してください。
- コードが適切に並列化されているかどうかを解析します。詳細は 47 ページの「コード中のボトルネックの検出」を参照してください。
- 無効な共有がないかどうかをチェックします。詳細は 47 ページの「無効な共有の修復」を参照してください。
- データの配置をチューニングします。詳細は 48 ページの「`dplace` と `runon` の利用」を参照してください。
- チューニングを支援する環境変数を設定します。詳細は 49 ページの「環境変数によるパフォーマンス・チューニング」を参照してください。

コードの並列化

マルチプロセッサ環境でのパフォーマンス・チューニングで最初に行うことは、チューニングに使用する並列化手法を選択することです。次の中から選べます。

- SGI Message Passing Toolkit (MPT)によるメッセージ交換インターフェイス(MPI: Message Passing Interface) を使う方法。この MPI は一般的な MPI ライブラリに比べて、SGI LX 3000 Series システム用に最適化されています。SGI LX 3000 アーキテクチャと SGI Linux NUMA の特徴が活用できます。

  MPI を使用するには `-lmpi` コンパイラ・オプションを使います。サポートされる環境変数の一覧は `mpi` マン・ページを参照してください。環境変数の一部は IRIX システムでのみ有効です（マン・ページに注記されています)。

  SGI MPT version 1.6.1 以降の `MPIO_DIRECT_READ` と `MPIO_DIRECT_WRITE` は Linux でローカル XFS システムに対して使用できます。

- OpenMP を使用する方法。C、C++、Fortran コードで OpenMP ディレクティブを使用する場合、次のコンパイラ・オプションが利用できます。

  `efc -openmp` / `ecc -openmp`
  : Intel コンパイラに組み込まれている Intel OpenMP フロントエンドを利用します。生成される実行ファイルは Intel OpenMP ランタイム・ライブラリ `libguide.so` への呼出しを行います。

  `guide`
  : Intel コンパイラで OpenMP コードを使用するもう一つの方法です。`guidec` (`ecc` の替わりに)、`guideefc` (`efc` の替わりに)、または `guidec++` によって、OpenMP ディレクティブを含むコードを `libguide` 呼出しにコンパイルします。詳細は 53 ページの「その他のパフォーマンス・ツール」を参照してください。

  `-openmp` オプションによるコンパイルが、Intel が推奨する Linux OpenMP コンパイラ手法です。この方法でパフォーマンスに問題がある場合は、`guide` によって改善される可能性があります。

- コンパイラの自動並列化機能を利用する方法。`efc` / `ecc` コンパイラで `-parallel` オプション、`-par_report` オプションを使用します。これらのオプションは並列化されたループを表示し、また、並列化されなかったループについてはなぜされなかったのかが説明されます。ソース・ファイルに多数のループが含まれるときは、`-override_limits` フラグを使って自動並列化を促進する必要があります。`-parallel` によって生成されるコードは OpenMP API を利用しており、標準的な OpenMP 環境変数や Intel 拡張環境変数が有効です。

コードの並列化度を見積もります。並列化度は次の式で計算できます。

```
p=N(T(1)-T(N)) / T(1)(N-1)
```

ここで T(1) はコードがシングル CPU で実行される場合の実行時間、T(N) は N 個の CPU で実行される場合の時間です。速度比は *speed-up*=T(1)/T(N) で求められます。

speed-up/`N` が 50% 未満の場合（`N>(2-p)/(1-p)` の場合）、CPU 数の追加は中断してスケーラビリティ改善のためのチューニングを検討してください。

CPU の活動度は `top`、`vmstat` などのコマンドや、PCP ツールを利用（たとえば `pmval kernel.percpu.cpu.user`）するか、あるいは PCP のビジュアル化機能を使ってリモート IRIX ワークステーションから確認できます。ハードウェア情報は `/proc/cpuinfo` や `/proc/meminfo` で確認できます。キャッシュ等の詳細情報は `/proc/pal/cpu`*X*（X は CPU 番号）で確認できます。

コード中のボトルネックの検出

コード中のボトルネックを検出できる各種ツールについての詳細は、50 ページの「プロファイラとパフォーマンス・ツール」を参照してください

無効な共有の修復

並列化後のプログラムが並列化前よりも遅い場合は、無効な共有が発生している可能性があります。無効な共有は、プロセッサのデータ・キャッシュの同じキャッシュ・ラインに対応する 2 つまたはそれ以上のデータ・アイテムが、共有メモリ・アプリケーションの複数のスレッドによってアクセスされない場合に発生します。異なる CPU 上で実行されている 2 つのスレッドが同じキャッシュ・ラインを変更しようとすると、キャッシュ・ラインは正確な内容を保ったまま複数の CPU 上で常駐できなくなり、一貫性を保つためにメモリ・サブシステムに移動させられます。これはアプリケーションのパフォーマンスとスケーラビリティの低下を招きます。データ・アイテムの書込みがなく読出しのみの場合は、キャッシュ・ラインは関与するすべての CPU 間で共有状態のまま残ります。無効な共有は、共有アレイの中で隣接する要素が異なるスレッドによって変更されたときに発生することがあります。

無効な共有が発生しているかどうかは、次のいくつかの方法で検出できます。

- パフォーマンス・モニタ `pfmon` の出力で、`BUS_MEM_READ_BRIL_SELF` イベントと `BUS_RD_INVAL_ALL_HITM` イベントを監視します。
- `pfmon` を使って `DEAR` イベントを監視し、キャッシュ・ラインを追跡します。
- PCP `pmshub` ユーティリティを使って、キャッシュ・トラフィックと CPU 利用率をモニタリングします。

メモ： `pmshub` ユーティリティは初期リリースには同梱されない可能性があります。最新情報はリリース・ノートを参照してください。

無効な共有が問題を引き起こしている場合は、次の方法で解決を試みます。

- HW カウンタを使ったプロファイルを実行して、キャッシュ・ライン共有のための記憶領域をモニタリングします。これによって、問題が発生している場所がわかります。
- データ構造またはアルゴリズムを変えます。

- 共有オブジェクトのプライベート / パブリック変数、共有データ、静的変数、コモン・ブロックを調べます。
- クリティカル・リージョンを使って問題を引き起こしているコード部分を特定します。

dplace と runon の利用

dplace コマンドは、複数のプロセスを指定された CPU 群にラウンドロビン方式でバインドします。バインドされたプロセスは移行されることはありません。この動作は IRIX システムの DSM_MUSTRUN と同様です。dplace のナンバリングは現在の CPU メモリ・セットのコンテキストで決まります。

runon コマンドは、指定されたリスト中の CPU にプロセスの実行を制限します。ただし、プロセスはリストされた CPU 間で自由に移動できます。

これらのコマンドの詳細については、以下の各節を参照してください。

MPI コードに対する dplace の使用

CPU とデータの親和性を高める最善の配置方法は、次のいずれかの手法を用いることです。

- Cpuset で MPI_DSM_MUSTRUN を使用します。
- MPI_DSM_CPULIST 環境変数を設定します。
- dplace-s1 コマンドを使用します。

SGI の MPI コンパイルされたコードを使用するときは、dplace のフル・パスと -s1 オプションを指定します。例を次に示します。

```
% mpirun -np 32 /usr/bin/dplace -c16-47 -s1 ./a.out
```

-s1 オプションで、dplace が軽量 MPI 管理プロセス (MPI sepherd process) を配置する手順をスキップします。

OpenMP コードに対する dplace の使用

Intel が提供する OpenMP コードを使用する場合は -x6 オプションを使用します。例を次に示します。

```
% setenv OMP_NUM_THREADS 4
% dplace -x6 -c5-8 ./a.out
```

-x6 オプションで、dplace が 2 つの軽量 OpenMP 管理プロセスを配置する手順をスキップします。

dplace と runon の違いに注意してください。

```
% setenv OMP_NUM_THREADS 4
% dplace -x6 -c5-8 ./a.out
% runon 5-8 ./a.out
```

dplace コマンドは管理プロセスの配置をスキップできるのに対して、runon はスキップできません。dplace コマンドはまた、各スレッドを特定の CPU に固定するのに対して、runon ではスレッドを CPU 5-8 の間に限定しますが、これらの CPU の範囲内では自由に移動できます。

dplace の詳細と使用法については、『Linux Resource Administration Guide』を参照してください。

環境変数によるパフォーマンス・チューニング

パフォーマンス・チューニングのためにいくつかの環境変数が利用できます。MPI の動作を制御する環境変数の詳細については、mpi(1) マン・ページを参照してください。

OpenMP 環境変数は、OpenMP ライブラリの動作に影響します。たとえばいくつかの環境変数は、アプリケーション中ですることがないスレッドや、他のスレッドとの同期のため待機中のスレッドの動作を制御します。他の変数は、OpenMP ライブラリのスレッド間でのループ繰返しスケジュールを設定します。次の環境変数は OpenMP 標準で定義されます。

- OMP_NUM_THREADS(デフォルトはシステム中の CPU 数)
- OMP_SCHEDULE(デフォルトは static)
- OMP_DYNAMIC(デフォルトは false)
- OMP_NESTED(デフォルトは false)

上記の環境変数以外にも、Intel による OpenMP 拡張が提供されています。次の 2 つは KMP_LIBRARY 変数による制御の例です。

KMP_LIBRARY 変数はランタイム実行モードを次のように設定します。

- serial に設定すると、シングルプロセッサ実行が適用されます。
- throughput に設定すると、処理待機中は CPU を他のプロセスに渡します。これはデフォルトの動作で、マルチユーザ環境でのシステム全体のパフォーマンスを維持します。IRIX での _DSM_WAIT=YIELD と同等の設定になります。
- turnaround に設定すると、待機処理中も CPU を明け渡しません。IRIX での _DSM_WAIT=SPIN と同等の設定になります。KMP_LIBRARY を turnaround に設定すると、他のユーザとの CPU リソース競合がない専用システムで実行されるベンチマークのパフォーマンスを改善できます。

プログラムの実行直後にセグメンテーション違反が発生するときは、KMP_STACKSIZE の値を増やしてみてください。これはスレッドのプライベート・スタック・サイズです。デフォルトは 4MB です。シェルのスタックサイズ・リミットも上げる必要がある場合があります。

プロファイラとパフォーマンス・ツール

パフォーマンス改善のポイントを特定するために使用できるツールがあります。以下にいくつかのツールの概要を説明します。

pfmon を使ったプロファイリング

pfmon ツールは Linux 用に設計されたパフォーマンス・モニタ・ツールです。Itanium Performance Monitoring Unit (PMU) を使って、バイナリを加工せずにカウント、サンプリングします。次のタスクが実行できます。

- 未加工のバイナリのプロセス単位でのモニタリング。
- システム全体のモニタリング・セッションの実行。任意の CPU で実行されている全プロセスのモニタリングが可能です。
- 専用 CPU 上でのシステム全体セッションの起動。複数 CPU 上で並列実行することも可能です。
- ユーザ・レベルまたはカーネル・レベルを指定したモニタリング。
- 基本イベント・カウントの集計。

- プログラムまたはシステムのサンプリング。同時に 4 個のイベントをモニタリングできます。

可能なオプションの一覧を見るには、pfmon -help コマンドを実行してください。

profile.pl スクリプト

profile.pl スクリプトは dplace の実行を含む、ユーザ・プログラム全体のプロファイリング処理を操作します。典型的な利用法は次の通りです。

```
% profile.pl -c0-3 -x6 command args
```

このスクリプトはプロセッサ 0-3 を指定します。-x6 は OpenMP コードを使用する場合のみ必要です。

CPU_CYCLES PMU イベントに対するプロファイルが *profile.out* に出力されます。このスクリプトでは、IA64_INST_RETIRED や L3_MISSES など他のイベントに対するプロファイリングも行えます。PMU イベントの一覧は pfmon -l の出力を参照してください。このスクリプトはパフォーマンス・モニタの下でのコマンド実行、実行ファイルとダイナミック・ライブラリにあるシンボル名とアドレスの対応づけをするマップ・ファイルの生成、プロファイル・アナライザの実行などもサポートします。

詳細は profile.pl(1)、analyze.pl(1)、makemap.pl(1) のマン・ページを参照してください。

MPI プログラムでの profile.pl の使用

MPI プログラムでは profile.pl スクリプトに -s1 オプションを付けて実行してください。例を次に示します。

```
% mpirun -np 4 profile.pl -s1 -c0-3 test_prog </dev/null
```

/dev/null を指定することで、MPI プログラムは TTY 入力を待たずにバックグラウンド実行されます。

VTune を利用したリモート・サンプリング

Intel VTuneパフォーマンス・アナライザはリモート・サンプリングによる試験を行います。VTune データ・コレクタは Linux システムで実行され、これに対応する GUI は IA32 Windows マシンで結果を解析するために用いられます。

VTune の詳細は下記 URL を参照してください。

```
http://developer.intel.com/software/products/vtune/vtune61/index.htm
```

メモ：VTune は初期リリースには同梱されない可能性があります。最新情報はリリース・ノートを参照してください。

GuideView の利用

GuideView はプログラムの並列実行に関するパフォーマンス情報をグラフィック表示します。GuideView は KAP/Pro Toolset の一部です。KAP/Pro Toolset はこの他に Guide OpenMP コンパイラと Assure Thread Analyzer を含んでいます。GuideView はデフォルトのソフトウェア・インストールではシステムにインストールされません。

GuideView は並列処理の問題点となるパフォーマンス・ボトルネックを、色を使って視覚的に示します。各プロセッサの動作状況は、階層化されたサマリによって詳細度別に報告されます。

統計データは対応するサマリを展開すると表示されます。サマリには、注目すべき場所に関する情報が含まれています（たとえば、全体のパフォーマンスに最もインパクトを与えるコード部分が示されます）。

プログラミングに有効な統計データを収集するには、コンパイラ・オプション `-O3`、`-openmp`、`-openmp_profile` を指定します。これによってデフォルト `libguide.a` の替わりに `libguide_stats.a` がリンクされます。次の例は、*swim* というファイルを生成するコンパイラ・コマンド行を示しています。

```
% efc -O3 -openmp -openmp_profile -o swim swim.f
```

プロファイリング・データを収集するには、このプログラムを次のように実行します。

```
% export OMP_NUM_THREADS=8
% ./swim < swim.in
```

プログラムが終了すると、GuideView で使用できる *swim.gvs* というファイルが生成されます。GuideView をこのファイルと共に起動するには、次のコマンドを実行します。

```
% guideview -jpath=your_path_to_Java -mhz=998 ./swim.gvs.
```

GuideView のグラフィック部分は Java で記述されています。Java 1.1.6-8 および Java 1.2.2 がサポートされており、これ以降のバージョンでも動作すると予想されます。Java がない場合は GuideView の機能は大幅に制限されますが、テキスト出力によって情報を得ることができます。テキスト出力の一部を次に示します。

```
Program execution time (in seconds):
cpu             :            0.07 sec
elapsed         :           69.48 sec
  serial        :            0.96 sec
  parallel      :           68.52 sec
cpu percent     :            0.10 %
end

Summary over all regions (has 4 threads):
# Thread                       #0       #1        #2       #3
  Sum Parallel         :   68.304   68.230   68.240    68.185
  Sum Imbalance        :    1.020    0.592    0.892     0.838
  Sum Critical Section:    0.011    0.022    0.021     0.024
  Sum Sequential       :    0.011  4.4e-03  4.6e-03   1.6e-03
  Min Parallel         : -5.1e-04 -5.1e-04  4.2e-04  -5.2e-04
  Max Parallel         :    0.090    0.090    0.090     0.090
  Max Imbalance        :    0.036    0.087    0.087     0.087
  Max Critical Section:  4.6e-05  9.8e-04  6.0e-05   9.8e-04
  Max Sequential       :  9.8e-04  9.8e-04  9.8e-04   9.8e-04
end
```

その他のパフォーマンス・ツール

次のパフォーマンス・ツールも、コードの最適化に有用です。

- *Guide OpenMP Compiler* は C、C++、Fortran に対する Intel の OpenMP 実装です。
- Intel の *Assure Thread Analyzer* はスレッド化アプリケーションのプログラミング・エラーをレコーディング無しで検出します。

これらの製品の詳細については、下記 URL を参照してください。

http://developer.intel.com/software/products/threadtool.htm

メモ：これらの製品は、SGI 製品上での完全検証は行われていません。SGI は、サード・パーティ製品の動作や合目的性に対する保証は行いません。

索引

A

Application performance measuring tools 18
Array Services 18

C

CD からブート 7、13
CD の内容 2
CpuMemSets のサポート 21
CSA (Comprehensive System Accounting) のサポート 22
CSA のサポート 22

D

dlook コマンド 34
dplace コマンド 34

E

EFI (Extended Firmware Interface) のサポート 30
EFI のサポート 30
Electric Fence デバッガ 39
ELF ヘッダの変更 44
Ethernet デバイス名 26

F

FLEXlm のサポート 32

G

gdb ツール 37
GNU デバッガ 37

H

HPC サポート
- Message Passing Toolkit 29
- Performance Co-Pilot (PCP) 29
- ライブラリとツール 29

I

idb ツール 38
I/O サブシステム
- HPC システムのための 24
- XFS ファイルシステムのサポート 28
- XSCSI のサポート 27
- XVM のサポート 28

IP アドレスの設定 8

L

L1/L2 コントローラ　33

M

Message Passing Toolkit のサポート　29

N

NUMA Tools　34
  dlook コマンド　34
  dplace コマンド　34

O

OS 拡張
  CSA(Comprehensive System Accounting)　22
  CpuMemSets　21
  Cpuset のサポート　22
  HPC 環境のための　21
  システム・パーティション　23
OS 構成の設定　1

P

PCP のサポート　29
Performance Co-Pilot のサポート　29
PROM のサポート　30

R

Rescue モード　12
root パスワード　9

S

SGIconsole のサポート　33
SystemController Software CD　11

X

XFS ファイルシステムのサポート　28
XSCSI デバイス命名法　27
XSCSI のサポート　27
XVM のサポート　28

い

インストール
  CD　10
  Open/Free Source Software　10
  Proprietary Software　10
  SGI Linux Environment 7.2 updates　11
  SystemController Software　11
  アップグレード　12
  インストール手法の選択　8、13
  概要　4
  再起動　10
  タイムゾーン　9
  追加ユーザ　9
  パスワード設定　9
  パッケージ・グループの選択　9
  パッケージの選択　10
  ベース OS SGI Linux Environment 7.2　6
  ユーザの作成　9

ログ　10
SGI ProPack for Linux　10

お

オープン・ソース・サイト　4

か

画面
Complete　10
Installation to Begin　10
Package Group Seleciton　9、10
Package Installation　10
SGI ProPack Welcome　10
Welcome　11

け

言語の選択　7、13

こ

固定命名
Ethernet デバイス　26
XSCSI デバイス　27
バス番号　24
コンパイラ　36

さ

サポートされる製品　17
サポート対象外　20

し

システム管理
EFI　30
FLEXlm　32
PROM チップ　30
SGIconsole　33
システム・コントローラ・ファームウェア　33
ツール　30
システム・コントローラ・ファームウェアのサポート　33
自動パーティション・オプション　8

せ

製品の一覧　17

そ

ソフトウェア
計画　5
インストール　5
ソフトウェアのアップグレード　12

ち

チューニング
chatr の使用　44
ddd ツール　38
dplace　48
Electric Fence　39
環境変数　49
偽共有　47
キャッシュ・パフォーマンス　43
コンパイラ・オプション　42

システム構成　35
出力の正確性の確認　37
シングルプロセッサ・コード　36
推奨コンパイラ　36
数値演算関数の利用　41
デバッグ・ツール
　Electric Fence　39
　gbd　37
　ibd　38
ヒープ破壊　39
プロファイリング
　dplace コマンド　51
　GuideView　52
　pfmon　50
　profile.pl スクリプト　51
　VTune アナライザ　52
並列化　45
ボトルネックの検出　47
マルチプロセッサ・コード　45
無効な共有　47
メモリ管理　43

て

ディスク・パーティション表　6

は

パーティションのサポート　23
パーティションの消去　8
パーティションの選択　8
ハードウェア・プラットフォーム　1
バス固定ナンバリング　24
バス番号　24
パスワードの必要性　5
パッケージ・グループの選択　9、10
パフォーマンスのチューニング　35

ふ

ファイアウォールの設定　9
ファイルシステムの修復　12
ファイル設定と LKCD　6
ファイルの設定　6
ファイルの配置　6

ほ

ボタン
　戻る　7

り

リリースの内容　1