2004－ARC－157　(8)
2004－HPC－ 97　(8)
2004／3／1

An MPICH-G Network on SuperSINET and its Performance

佐藤周行[†1]　南里豪志[†2]　長尾光悦[†3]
高井昌彰[†3]　平野彰雄[†4]

SuperSINET 上の Grid 環境で MPICH-G のネットワークを構築した。MPI 環境の提供は MPI ユーザの計算 Grid への移行に対するバリアを低くすることができる。また, MPI は並列計算機環境と分散環境を同時に記述できる計算モデルを構築するための単純化にも有用である。本稿では SuperSINET の Grid 環境での MPICH-G ネットワークの構築とその性能評価, さらにそれをもとにした LongFat Network 向けの性能改善について報告する。筆者らの研究環境の現状では処理オーバーヘッドは満足できること, バンド幅に関しては window size の拡張の効果が大きく, メッセージ長の大きなところでは満足できる結果を示すことがわかった。

An MPICH-G Network on SuperSINET and its Performance

SATO Hiroyuki,[†1] NANRI Takeshi,[†2] NAGAO Mitsuyoshi,[†3]
TAKAI Yoshiaki[†3] and HIRANO Akio[†4]

The MPI environment can lower the barrier for an MPI programmer to move to computational Grid. Moreover, MPI programming model is useful in providing a simple performance model for both parallel environment and distributed environment. In this paper, we report our MPICH-G network built on SuperSINET Grid Environment, together with its performance and longfat-network-oriented performance improvement. In our environment, it is showed that the overhead is low enough, and that the window size enlargement is useful for exploiting bandwidth of MPICH-G, and the bandwidh utilization shows a good result in large messages.

1. はじめに

Grid という形で分散環境の本格的な展開がはじまったのは日本ではここ数年のことである。SuperSINET[3] は全国を 10Gbps の幹線で結ぶネットワークのプラットフォームであり, いわゆる国立大学の情報基盤センター群にも接続がのびている。そのなかの各端末とは 1Gbps のラインで接続されている。

Grid を大規模計算のために使うにはさらに計算モデルとそのためのミドルウェアが提供されていなければならない。われわれは代表的なミドルウェアをインストールして, その運用を含めたテストを行っている。

我々の研究のプラットフォームとなる SuperSINET の Grid 実験線において 2004 年 1 月現在, 東大情報基盤センターからみたネットワークの連結性/globus の連結性/MPICH-G2 の連結性は図 1 のようにまとめられる☆。

このうち, 重要なのは MPICH-G2 による連結性である。MPI は, 並列環境で大規模計算を行ってきたユーザが Grid という本質的にヘテロな環境で計算を展開するときの中核的な通信モデルを提供することができる。計算を主目的とする Grid(計算 Grid) では, MPI 環境の提供は MPI ユーザの移行に対するバリアを低くすることができる。また, MPI は並列計算機環境と分散環境を同時に記述できる計算モデルを構築するための単純化にも有用である。

計算 Grid では, 連結性とともに性能面での評価が必須である。本稿はこれまでに構築された環境で MPICH-G2 の性能をレポートし, あわせて Super-

†1 東京大学 情報基盤センター
Information Technology Center, the University of Tokyo
†2 九州大学 情報基盤センター
Computing and Communications Center, Kyushu University
†3 北海道大学 情報基盤センター
Information Initiative Center, Hokkaido University
†4 京都大学 学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

☆ 他センターからみればまた別の図になることはもちろんである。

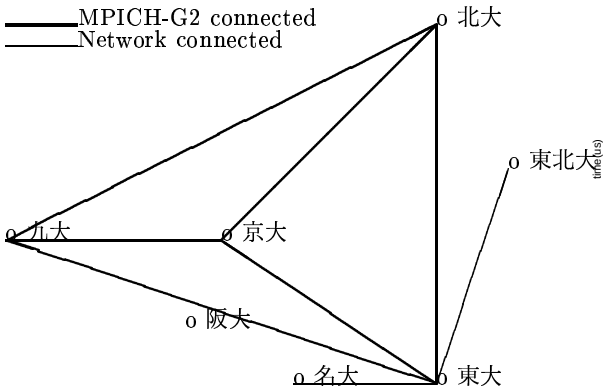


図 1 東大からみた SuperSINET Grid 実験線での連結性
ここで MPICH-G2 connected とは東大から当該 2 サイトに MPICH-G2 のプロセスが生成され, それらが通信可能であることとする。

SINET という LongFat Network 環境でバンド幅を使いきるための改善手法の適用についても報告する。

2. MPICH-G2 Platform

現在東大情報基盤センターは北大, 京大, 九大と MPICH-G2 connected である。各サイトの環境は表 1–3 のとおりである。

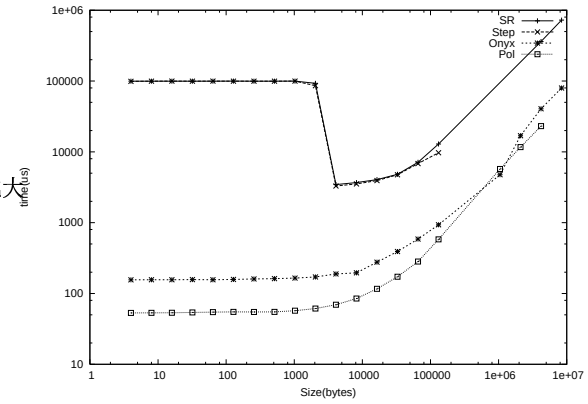
3. Performance of MPICH-G2

MPICH パッケージ添付の mpptest を使って 2 地点間のブロッキング通信の性能を計測した。なお, 計測は他の負荷の影響を大きくうけるが, 測定環境として他のプロセスがほとんど存在しない環境が得られたので, 今回の測定においては他の負荷の影響は無視できると考えている。

3.1 Performance between Selves

SR8000, Onyx, PrimePower(Pollux) で 2 個 MPICH-G2 を立ち上げてそれらで通信性能を計測した (図 2)。なお, SR8000 は 8 プロセッサ構成になっているが, プロセスの実行モデルが他の SMP マシンと異るので単純な比較はできない。

図から, Onyx と PrimePower では処理のオーバーヘッドが小さいメッセージでの性能を悪くしていることが観察できる。しかし Long Fat Network という環境下ではほとんど無視できる値であることもわかる。大きなメッセージでは PrimpPower では 1Gbps の環境でもボトルネックになっていないこと, Onyx でもおおむねボトルネックになりにくいことが, また後述



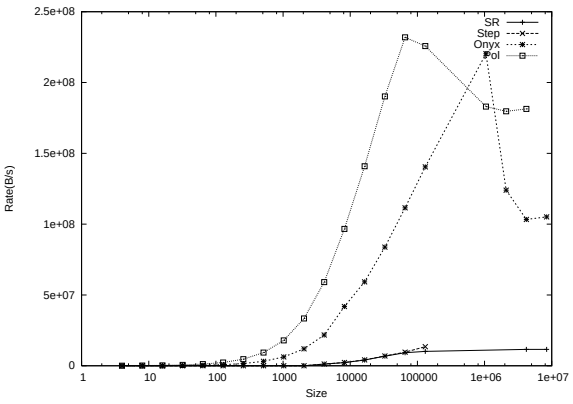


図 2 自分自身との通信性能: 経過時間 (上), 転送速度 (下)

の TCP の処理性能と併せ考えれば現状では全体として MPICH-G2 のプロセスが処理のボトルネックにならないことを確認した。

3.2 Performance across Sites

window size をデフォルトのままにして計測した。なお, 各マシンの投入時期の違いから以下については例外である。

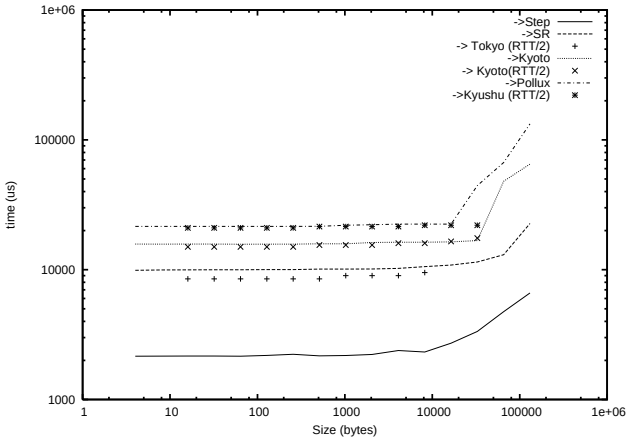
北大 ⟷ 京大, 九大　onyx の window size は 1MB
九大 ⟷ 京大　kyoto の window size は 1MB

計測結果を図 3 に示す。

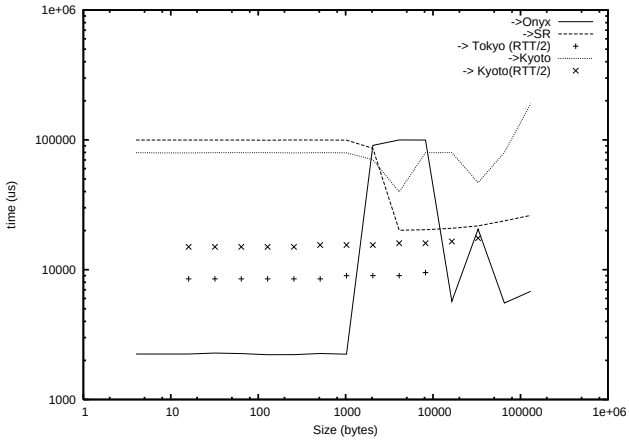
Onyx と PrimePower では, default の window size の前では, RTT/2 に近い時間で通信が可能になっていることがわかる。LongFat Network という環境下では満足できる結果である。

4. Exploiting Bandwidth

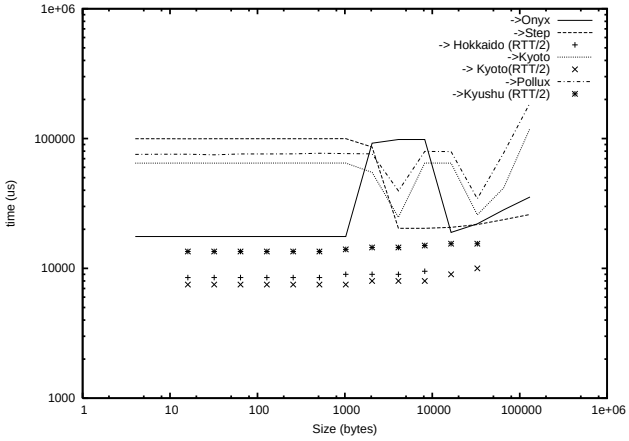
SuperSINET のような典型的な Long Fat Network



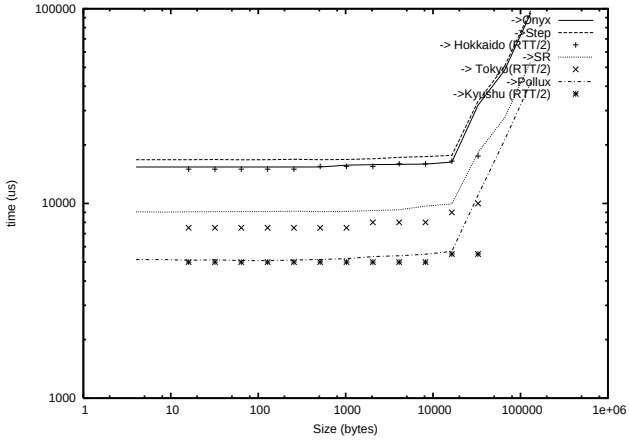
Onyx が sender



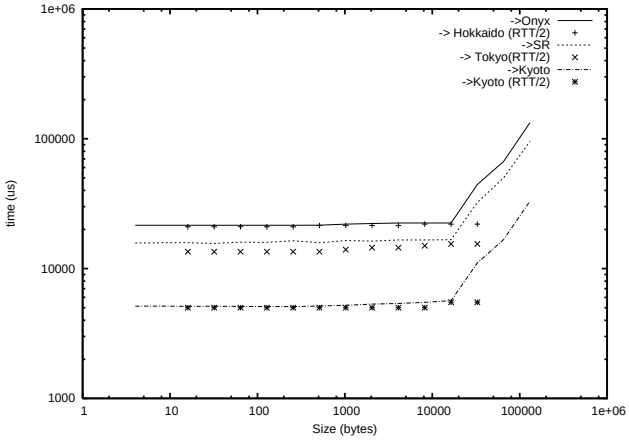
Step が sender



SR が sender



Kyoto が sender



Pollux が sender

図 3 各サイトからの通信性能 (経過時間)
(RTT/2) の項は各サイト間の RTT を 2 で割った時間

表 1　各サイトの端末

北大	SR8000-compact(step)	450MHz × 8 procs (Power3 compatible)
	SGI Onyx(onyx)	600MHz × 32procs (MIPS)
東大	SR8000-compact(sr)	450MHz × 8 procs (Power3 compatible)
京大	PrimePower(kyoto)	1.35GHz × 96 procs (SPARC64V)
九大	PrimePower(pollux)	1.35GHz × 16 procs (SPARC64V)

表 2　各サイト間の round trip time

北大 ⟷ 東大	17ms	北大 ⟷ 京大	30ms
北大 ⟷ 九大	42ms	東大 ⟷ 京大	15ms
東大 ⟷ 九大	27ms	京大 ⟷ 九大	10ms

表 3　各端末の Globus, MPICH-G2 のバージョン

	Globus	MPICH
step	Globus 2.0*	MPICH 1.2.5 G2
onyx	Globus 2.2	MPICH 1.2.5 G2
sr	Globus 2.0β	MPICH 1.2.5 G2
kyoto	Globus2.4.3	MPICH 1.2.5.2 G2
pollux	Globus 2.0*	MPICH 1.2.5.1a G2

*) ベンダ提供のものをそのまま使用。

の性能を引出すために先行プロジェクトではさまざまなことが試されている[1),4)]。その中で標準的な手法のひとつが TCP window size の拡張 (RFC1323) と SACK(RFC2018) である。

実験環境では東大をのぞく各サイトで RFC1323 と RFC2018 が利用可能になっている。表 4 に示す値まで window size を拡張して, 特に大きいメッセージのやりとりについての性能を測定した。また, RFC2018 については, Onyx では default で On であり, Solaris では send/receive ともに On に変更して測定している。図 4 に転送速度の結果を表す。

図 4 では, default の環境ではメッセージのサイズは 128KB まで, window size の変更後は 8MB までのメッセージサイズで性能を測定した (図中 LL)。

性能の差が観察されるのは, window size の変更前後からである。これから, RFC1323 によるチューニングが効果的であることが観察される。

5. Performance Indices

ここでは以下の指標を定義し, 各サイト間のリンクについてその指標にもとづいた評価を行う。

定義 1　MPI の処理のオーバーヘッドを評価するために OH を以下で定義する。

$$OH_s = \frac{(\text{size } s \text{にかかる RTT}/2)}{(\text{size } s \text{ の MPI メッセージ通信にかかる時間})}$$

定義 2　MPI のバンド幅の性能をみるために BW を以下で定義する。
MPI でのバンド幅 を

$$\frac{s}{\text{size } s \text{ の MPI メッセージ通信にかかる時間}}$$

として,

$$BW_s = \frac{\text{MPI でのバンド幅}}{\text{TCP でのバンド幅}}$$

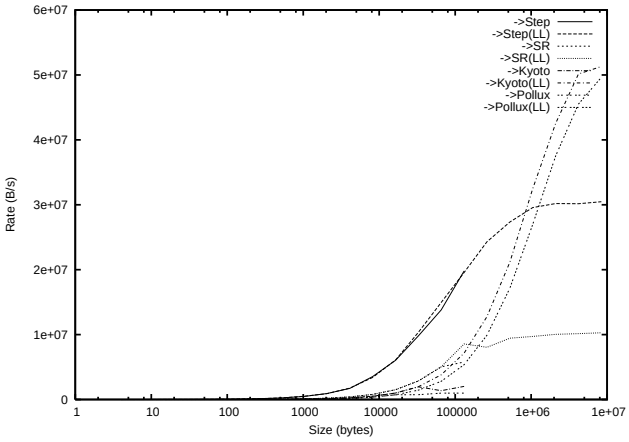
TCP でのバンド幅は `iperf` で計測したものを用いる。参考のために, TCP の処理性能 TCP を便宜的に以下で定義する。これに責任をもつのは端末のハードウェア, システムソフトウェアとネットワークである。

定義 3

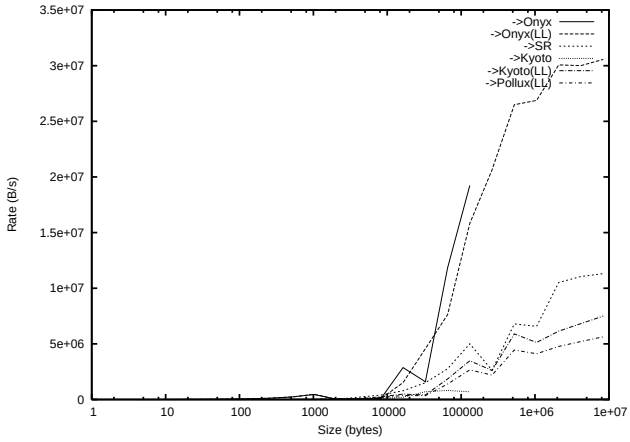
$$TCP = \frac{\texttt{iperf} \text{ で計測した TCP のバンド幅}}{\text{理論性能}}$$

表 5 に, 各接続 (北大の場合は Onyx) について, OH については OH_{64B}, OH_{64KB}, BW については BW_{64KB}, BW_{8MB} の値を示す。なお TCP の計算中使用した TCP の理論性能はいずれの場合も 1Gbps である。またこの値は window size について数点について計測してその最大値を取るというごく荒い決め方をしている。

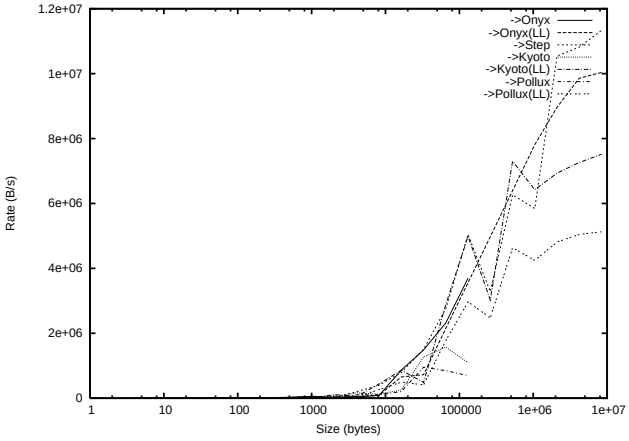
MPICH-G2 のオーバーヘッドは, OH をみると, Onyx と PrimePower(kyoto,pollux) では満足出来る値であることが観察できる。また BW をみると, バンド幅については東大との接続をのぞき, BW_{8MB} は満足できる値を示すことが観察できる。また, BW_{64KB}



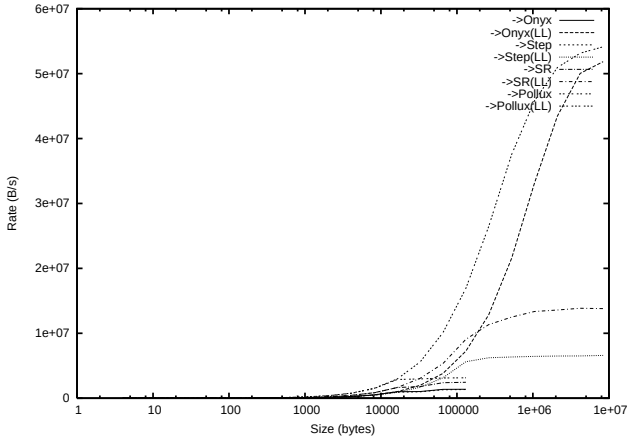
Onyx が sender



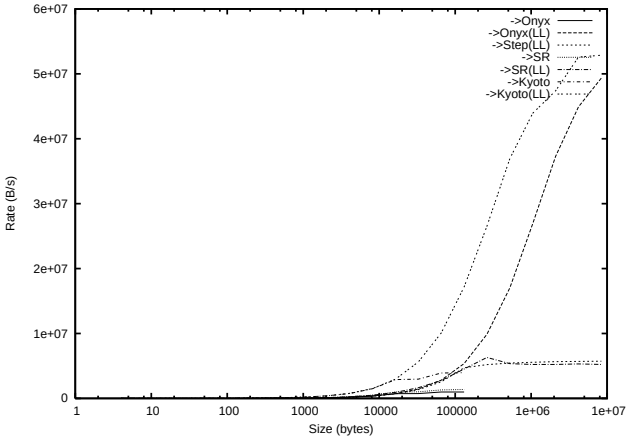
Step が sender



SR が sender



Kyoto が sender



Pollux が sender

図 4 チューニング後の各サイトからの通信性能 (Rate)
(LL) の項は各マシンでできるだけのチューニングを行った時の測定結果を示す

表 4　RCP1323 にしたがって拡張した TCP Window Size

	北大 Onyx*	東大	京大			九大		
(send 側)			対北大	対東大	対九大	対北大	対東大	対九大
default	60KB	256KB+	16KB			16KB		
拡張	1MB	–	3MB	1.5MB	1MB	5.25MB	3.375MB	1.25MB
(receive 側)			対北大	対東大	対九大	対北大	対東大	対九大
default	60KB	64KB+	24KB			24KB		
拡張	1MB	–	3MB	1.5MB	1MB	5.25MB	3.375MB	1.25MB

*) 北大 Onyx では, `MPICH_GLOBUS2_TCP_BUFFER_SIZE` の拡張が有効に動作するので (上の設定をしたうえで) これを 8.1MB に設定して測定を行った。
+) `iperf` の出力した値による。

表 5　性能指標

from⟶to	TCP	OH		BW		8MB での性能 (MB/s)
		64B	64KB	64KB	8MB	
北大 ⟶東大	0.089	0.851	0.655	0.454	0.924	10.277
⟶京大	0.520	0.973	0.870	0.058	0.789	51.309
⟶九大	0.398	0.975	0.900	0.057	0.997	49.642
東大 ⟶北大	0.089	0.483	0.279	0.194	0.901	10.028
⟶京大	0.111	0.116	0.330	0.208	0.542	7.518
⟶九大	0.081	0.178	0.366	0.175	0.505	5.127
京大 ⟶北大	0.520	0.975	0.873	0.059	0.798	51.844
⟶東大	0.111	0.824	0.611	0.385	0.995	13.803
⟶九大	0.549	0.994	0.770	0.147	0.789	54.133
九大 ⟶北大	0.398	0.975	0.899	0.056	0.992	49.347
⟶東大	0.081	0.844	0.568	0.272	0.516	5.233
⟶京大	0.549	0.991	0.768	0.147	0.770	52.869

註) OH, BW ともに window size をチューンしたあとの値

の値は低いが OH_{64KB} をみれば, 改善の余地は少ないことが観察できる。

6. Concluding Remarks

われわれは様々な大規模数値計算プログラムを MPICH-G2 network 上で動かすことを計画しており, そのための性能モデル構築のために今回のデータ計測は必須であった。今回の計測と OH, BW の観察により, MPICH-G2 の性能の良さがわかった。

大規模計算を行うときに predictive な性能モデルを構築することは必須である (e.g. Kerbyson[2])。性能に影響する要素としてはこの他に同期や bisection bandwidth などがあり, 自明でない計算モデルの構築が求められている。今後, これらについても性質を明かにしてこれらのデータを用いて Metacomputing 用の性能モデルを構築することを計画している。

また TCP の改善も大きな課題として残った。早急に原因を調査して対応したい。

参 考 文 献

1) Hacker, T.J., Athey B.D., Sommerfield J.: "Experiences Using Web100 for End-To-End Network Performance Tuning," `http://www.web100.org/docs`, 2002.
2) Kerbyson, D.J., Alme, H.J., Hoisie, A., Pertini, F., Wasserman, H.J., Gittings, M., "Predictive Performance and Scalability Modeling of a Large-Scale Application," SC2001, 2001.
3) `http://www.sinet.ad.jp`
4) Enabling High Performance Data Transfers, `http://www.psc.edu/networking/perf_tune.html`, 1998.