情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2011-HPC-129 No.5
2011/3/15

分散ファイルシステムの性能監視とボトルネック特定

大岸智彦[†]　西谷明彦[†]　大辻弘貴[††]　建部修見[†††]

近年，大容量のデータを一元管理できる分散ファイルシステムへの期待が高まっている．一方で，分散ファイルシステムは，複数のネットワークやサーバで構成されたシステムであるため，ボトルネック箇所や要因を特定しにくいという問題がある．本稿では，ボトルネックが存在する場合の分散ファイルシステムの性能を評価するとともに，分散ファイルシステム上の各サーバの通信ログを用いてボトルネック箇所を特定する手法について述べる．また，本特定結果を利用した分散サーバシステムの運用管理について考察する．

Performance Monitoring and Bottleneck Identification for a Distributed File System

Tomohiko Ogishi[†]　Akihiko Nishitani[†]
Hiroki Ohtsuji[††]　Osamu Tatebe[†††]

Recently, distributed file systems (DFSs) become promising technologies for their capability of managing a huge amount of data with a single server system. On the other hand, the location and the cause of bottleneck are difficult to identify since the resources are separated into multiple networks and servers. In this paper, the performance of a DFS is evaluated under several bottleneck situations and a proposed method that identifies the location of the bottleneck using communication logs collected by each server of the DFS is described. Furthermore, the operation and management methodology for a DFS using the information of identified bottleneck is discussed.

1. はじめに

近年，インターネット・イントラネット上で提供されるサービスや業務プロセスにおいてデータの大容量化が進んでおり，大容量のデータを一元管理できる分散ファイルシステムへの期待が高まっている．これまでに多くの分散ファイルシステム[1, 2, 3]が研究開発されてきたが，これらの共通点は，スペックが異なる複数の PC サーバをインターネットなどベストエフォートである場合も含めたネットワークで接続し，ネットワーク経由のプロトコルを用いて複数 PC サーバ上のストレージを単一のストレージを仮想化し，耐障害性やデータの一貫性などを保証するソフトウェアである．RAID 等のストレージを備えた高信頼なサーバを使う場合に比べ，利用目的に応じて，システム全体の信頼性やストレージ容量などを柔軟に選択できることが利点である．

一方，分散ファイルシステムは，複数のネットワークやサーバで構成されたシステムであるため，エンドユーザからのファイルの書き込みにおける性能劣化などの不具合が発生した場合に，例えばサーバの処理能力不足，ネットワーク遅延などの要因が考えられるものの，ボトルネック箇所や要因を特定しにくいという運用上の問題がある．この問題は，不具合発生時の対策の遅れにつながる恐れがある．

そこで筆者らは，分散ファイルシステム内のボトルネック箇所の特定手法について検討した．まず，サーバやネットワークのボトルネックが分散ファイルシステムに与える影響を評価するため，ボトルネックが存在する環境を試験的に構築し，ファイルの読み込み・書き込みのスループットを評価した．このスループットは，システム内のブロック転送の処理時間によって定められると仮定し，各サーバ上でブロック転送に関わる通信ログとして，イベント発生時刻，i ノード番号，ブロック転送の回数，総転送サイズ，ブロック転送処理時間などを収集する仕組みを実装した．各サーバの通信ログのうちブロック転送処理時間に着目し，その数値のサーバ間での比較により，ボトルネック箇所を特定できることを確認した．本解析の自動化により，分散ファイルシステムをベースとしたサーバシステムの運用管理ツールとして適用可能である．

本稿の構成を以下に述べる．次章では，本稿が対象とする分散ファイルシステムの構成と，性能の定義について述べる．３章では，分散ファイルシステム上で収集する通信ログの詳細について述べる．４章では，各ボトルネックを発生させるための実験方法と，実験結果としてボトルネックがスループットや各ブロック転送処理時間に与

[†] (株)KDDI 研究所
KDDI R&D Laboratories, Inc.
[††]筑波大学情報学群情報科学類
Collage of Information Science, University of Tsukuba
[†††]筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba

える影響について述べる．５章では実験結果を総合的に分析するとともに運用管理ツールとしての適用について述べ，６章で結論を述べる．

2. 分散ファイルシステムの構成と性能

分散ファイルシステムは，図１に示すように，ネットワーク上に存在する分散ファイルサーバ上のディスクを１つの仮想的なストレージとして構築したシステムである．通常，エンドユーザにサービスを提供する Web サーバ，ファイル共有サーバ等のアプリケーションサーバがローカルディスクや固定的なネットワークを経由して接続された NAS の代わりに利用する．アプリケーションサーバが動作するサーバ上では，分散ファイルシステムに接続するためのクライアントを動作させる．このサーバをプロキシサーバと呼ぶ．分散ファイルシステムは，複数のプロキシサーバ，複数の分散ファイルサーバが，ネットワーク経由(システムネットワークと呼ぶ)で接続されたシステムである．エンドユーザ，プロキシサーバ間を接続するユーザネットワークには，エンドユーザの宅内あるいは企業内のネットワーク, FTTH, ADSL などのアクセス回線，Web サービスやファイル共有サービスなどを提供する ASP 内のネットワークが含まれる．図には明記していないが，分散ファイルシステムには，一般に，ファイルの格納先サーバやファイル名などのメタデータ情報を管理するメタデータサーバも含まれる．

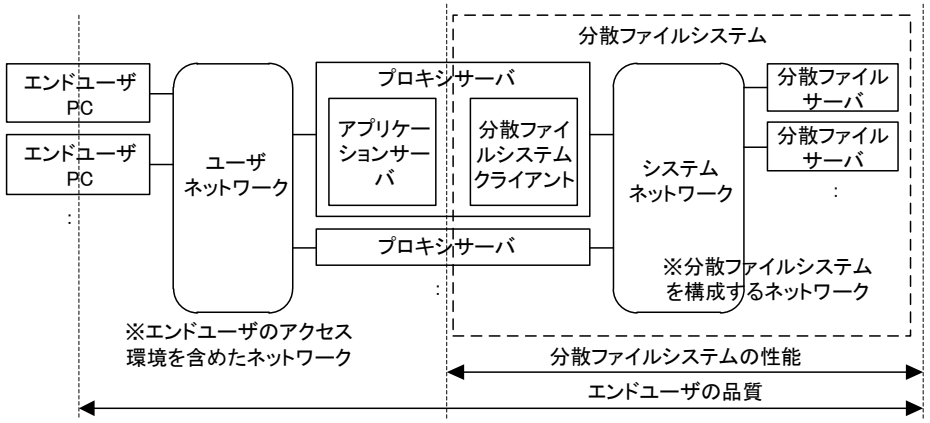

図１　分散ファイルシステムの構成

図１の構成において，分散ファイルシステムの性能は，a)分散ファイルシステムクライアント，b)システムネットワーク，c)分散ファイルサーバ，の性能によって決定される．また，エンドユーザの品質は，上記に d)ユーザネットワークの性能，を加えたもので決定される．

分散ファイルシステムをベースとしたアプリケーションサーバを運用するには，エンドユーザの品質低下に対して適切な改善策を施す必要がある．例えば，分散ファイルサーバに性能劣化があることが判明した場合には，分散ファイルサーバの数を増やす，あるいは，分散ファイルサーバを高性能なものに置き換えるなどの改善策を施す必要がある．従って，分散ファイルシステムの運用を行うにあたって，a)～d)の性能を個別に把握し，これらの性能値に異常があることを検知できることが重要と考える．

3. 通信ログの取得

分散ファイルシステムの構成要素毎の性能異常値を把握するには，分散ファイルシステムの構成要素である分散ファイルシステムクライアント（以降，クライアントと呼ぶ），分散ファイルサーバ上で通信ログを取得することが有用である．有用性の観点から，通信ログの取得において以下の要求条件を設定した．

- 通信ログの取得自体がサーバの性能に大きく影響を与えるものであってはならない．また，運用管理においては，通信ログをサーバ外部に転送し一箇所に集約する場合が考えられる．このため，通信ログは，CPU 負荷をかけずに簡易に取得できるものとし，また，必要最小限の情報のみを蓄積する．
- 品質劣化の原因が，個々のエンドユーザ，個々のプロキシサーバ，個々の分散ファイルサーバなどに特化している場合も考えられる．このような原因の切り分けが可能となるように，通信ログとして，エンドユーザやサーバ等を識別できる情報が必要である．
- エンドユーザからのファイルアクセス要求があった際，クライアント，分散ファイルサーバは，大部分のプロトコル処理時間をブロック転送に費やす．このブロック転送に関わる処理を的確に数値化することが重要である．

これまでに研究開発されてきた分散ファイルシステムにおいても，通信ログ相当のものを取得することは可能であるが，一般にはデバッグ目的で作成されたものであり，上記１つめの要求条件を満たしていない．そこで筆者らは，広域分散環境での利用を想定したオープンソースの分散ファイルシステム Gfarm[4]を用いて，ボトルネック特定に関する評価を行うこととし，通信ログ取得機能を実装した．具体的には，クライアント，分散ファイルサーバが動作するプロセスにおいて，メモリ上で通信状態を管

理し，ブロック転送処理の要求・応答のタイミングで時刻取得を行い，通信状態を更新する．ユーザからのファイルアクセス要求が完了したタイミングで通信状態をログファイルに追記する．従って，１ユーザからの１回のファイルアクセスに対して，１つの通信ログのレコードが作成されることになる．

表１に，クライアント，分散ファイルサーバで所得する通信ログの項目を示す．Gfarm ユーザ名は，Gfarm ファイルシステムを使用するユーザアカウントの名前である．エンドユーザと Gfarm ユーザを１対１でマッピングする場合には，個々のエンドユーザの識別が可能である．アクセス元 IP アドレスを利用する方法も考えられるが，NAT 配下の場合に区別できないなど不完全なため実装していない．ブロック転送処理は，書き込み要求 write()と読み込み要求 read()に分け，１回のファイルアクセスにおいて発生した回数(write()回数，read()回数)，各関数処理に費やした時間の合計(write()合計時間，read()合計時間)，書き込み・読み込み要求のブロックサイズの合計（書き込み総転送サイズ・読み込み総転送サイズ）を集計し，ファイルアクセス処理完了時に記録する．４章の実験評価において記載している処理時間とは，write()合計時間あるいは read()合計時間のことである．i ノード番号，世代番号の組は一意にファイルの同一性を識別するため，クライアントと分散ファイルサーバの通信ログの対応付けに利用する．

表１　通信ログの項目

サーバ種別	パラメータ
クライアント	イベント時刻, 自クライアント名, 接続先ファイルサーバ名, Gfarm ユーザ名, ファイル名, i ノード番号, 世代番号, write()回数, write()合計時間，書き込み総転送サイズ，read()回数，read()合計時間，読み込み総転送サイズ
分散ファイルサーバ	イベント時刻，接続元クライアント名，自分散ファイルサーバ名，Gfarm ユーザ名，i ノード番号，世代番号，write()回数，write()合計時間，書き込み総転送サイズ，read()回数，read()合計時間，読み込み総転送サイズ

4. 実験評価

本章では，実験環境の構成と，実験結果について述べる．

4.1 実験環境

図２に示すように，プロキシサーバ１台，ファイルサーバ４台，メタデータサーバ１台が 1Gbit/s の LAN 上に接続された分散ファイルシステムに対して，エンドユーザ４台が別の 1Gbit/s LAN を経由して接続する構成を構築した．プロキシサーバには２枚の NIC が搭載されており，それぞれ異なるネットワークに接続されている．メタデータサーバとそのデータを保管するデータベースサーバ(メタデータサーバ上で動作)については，本稿では性能評価の対象外としているが，Gfarm ファイルシステムにおける必須構成要素のため，過負荷等の影響が生じないような環境で導入している．

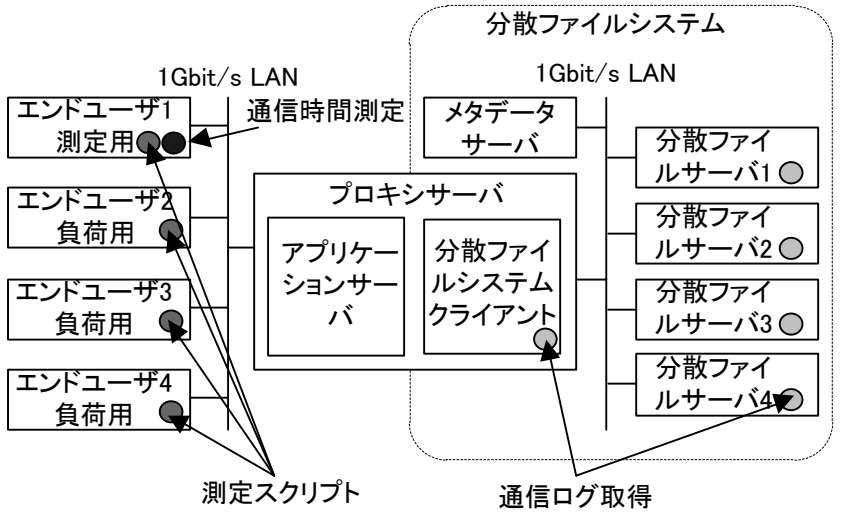

図２　実験環境の構成

実験方法の概要を以下に述べる．測定用アプリケーションとしては，Linux コマンドとしても実装されている dd および rcp (remote copy)を使用し，それぞれについてプロキシサーバ上のアプリケーションサーバとして samba サーバ，rcp サーバを動作した．これらの場合，FUSE[5]を経由して Gfarm ファイルシステムにアクセスする．dd では，エンドユーザ PC 上でのディスク I/O が発生しないように，書き込み時には/dev/zero からの読み込み，読み込み時には/dev/null への書き込みを行った．エンドユーザ PC のうち１台(エンドユーザ１)で通信時間を測定し，残り(エンドユーザ２～４)は負荷発生のみを行った．エンドユーザ PC は，測定スクリプトを用いて，sleep を入れることなく 10 回連続の書き込みあるいは読み込みを行い，このうちエンドユーザ 1 は 1 回毎に通信時間を測定した．読み込みの試験は，同等の条件の書き込みの

試験で保存されたデータを読み込むこととした．読み込み時にディスクキャッシュが効かないように，キャッシュを削除した上で測定を行った．

4.2 正常時

まず，ネットワークの劣化や，サーバの過負荷が無い正常時の評価を行った．エンドユーザ２～４は稼動させず，負荷が無い状態で，エンドユーザ１からのみ測定を行った．その結果を表２に示す．比較のため，samba や rcp を使用せず，プロキシサーバで dd を実行した結果も載せる． samba 使用時には，rcp と比べて書き込み，読み込みともスループットが小さいことが確認された．プロトコルのオーバーヘッドによるものと考えられる．

表２　正常時のスループット

アプリケーション	samba		rcp		プロキシサーバで dd	
	write	read	write	read	write	read
平均スループット (MB/s)	36.63	28.50	55.59	45.71	56.83	75.26
スループット変動 最小～最大(MB/s)	36.2～ 36.8	27.1～ 30.1	52.63～ 58.62	41.67～ 50.00	49.1～ 68.7	66.3～ 84.8

4.3 ユーザネットワーク (エンドユーザープロキシサーバ間)の劣化

まず，ユーザネットワークの劣化が与える影響について評価した．ネットワーク劣化を発生させるため，プロキシサーバのエンドユーザ側の NIC において，Linux コマンドである tc を用いて，遅延およびパケットロスを設定した．遅延は 0～100msec，パケットロスは 0～5%の範囲で設定し，それぞれ一方のみを変化させた．

遅延とスループット(10 回の測定の平均値)，パケットロスとスループットの関係を図３に示す. samba の場合, 遅延 5msec 以上において, 書き込み, 読み込みとも 10MB/s 未満となり，わずかな遅延においても著しくスループットが低下した．rcp では，この傾向は少し緩やかであり，遅延が 100msec に増えたときに，ようやく 10MB/s 程度に低下した．パケットロスに関しても，パケットロス率の増大に伴い，スループットが低下する傾向があったが, rcp 書き込みの場合はこの傾向は非常に緩やかであった．

次に，エンドユーザの通信時間(書き込み・読み込みの所要時間)(表中の end user)，クライアントのブロック転送処理時間(表中の client)，分散ファイルサーバのブロック転送処理時間(表中の file server)の比較結果を表３に示す．samba の読み込み時を除いて，ユーザネットワークに遅延やパケットロスが生じても，クライアントや分散ファイルサーバでのブロック転送処理時間にはほとんど変化がないことが分かる．

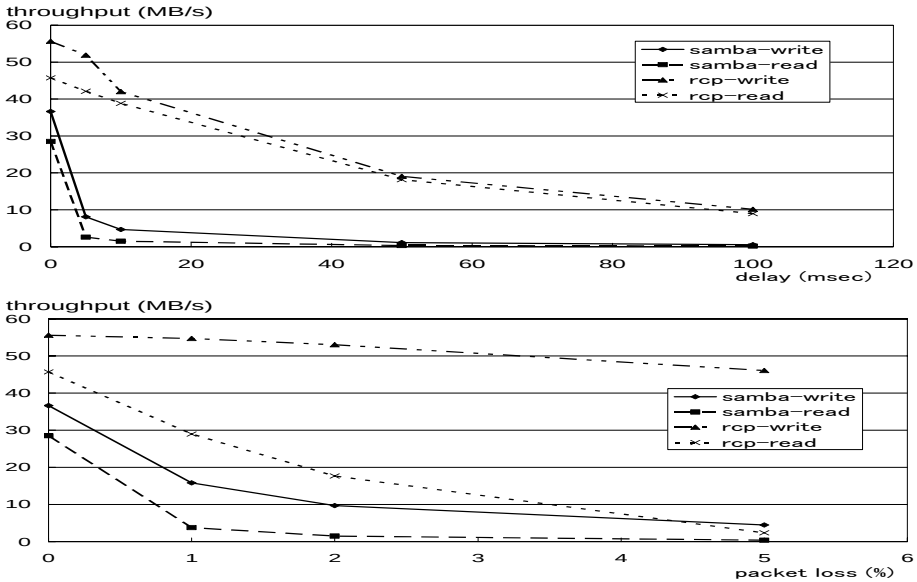

図３　ユーザネットワークの劣化とスループット

表３　ユーザネットワークの劣化とブロック転送処理時間

process time	delay	end user	client	file server
samba−write	0	2.864	1.129	0.208
	5	13.005	1.134	0.212
	10	22.430	1.127	0.208
	50	93.904	1.153	0.226
	100	185.209	1.171	0.231
samba−read	0	3.684	1.226	0.294
	5	39.984	1.820	0.891
	10	72.026	2.110	1.180
	50	327.326	3.135	2.193
	100	650.028	3.567	2.623
rcp−write	0	1.800	1.145	0.227
	5	1.930	1.137	0.217
	10	2.410	1.264	0.231
	50	5.250	1.154	0.235
	100	9.860	1.168	0.248
rcp−read	0	2.200	1.854	0.927
	5	2.380	1.827	0.900
	10	2.590	1.811	0.883
	50	5.530	1.814	0.886
	100	11.160	1.801	0.872

unit: delay (msec), end user, client, file server (msec)

process time	packet loss	end user	client	file server
samba−write	0	2.864	1.129	0.208
	1	6.738	1.132	0.209
	2	10.942	1.160	0.240
	5	23.678	1.151	0.230
samba−read	0	3.684	1.226	0.294
	1	28.220	1.874	0.944
	2	71.435	2.242	1.309
	5	275.629	3.151	2.209
rcp−write	0	1.800	1.145	0.227
	1	1.830	1.126	0.206
	2	1.890	1.170	0.231
	5	2.360	1.224	0.234
rcp−read	0	2.200	1.854	0.927
	1	3.480	1.771	0.844
	2	5.780	1.838	0.911
	5	42.930	1.991	1.062

unit: packet loss (%), end user, client, file server (msec)

4.4 システムネットワーク (プロキシサーバー分散ファイルサーバ間)の劣化

次に，システムネットワークの劣化が与える影響について評価した．ネットワーク劣化を発生させるため，プロキシサーバの分散ファイルサーバ側の NIC において，遅延およびパケットロスを設定した．同様に，遅延は 0～100msec，パケットロスは 0～5%の範囲で設定し，それぞれ一方のみを変化させた．

遅延とスループット，パケットロスとスループットの関係を図４に示す．ユーザネットワークの場合とは異なり，samba における遅延増加の伴うスループット低下が緩やかになり，rcp では遅延増加とともにスループットが大きく低下した．また，パケットロス率の増大に伴い，スループットが低下する傾向があったが，rcp 読み込みの場合はこの傾向が非常に緩やかであった．

次に，エンドユーザの通信時間，クライアントのブロック転送処理時間，分散ファイルサーバのブロック転送処理時間の比較結果を表４に示す．クライアントのブロック転送処理時間は，エンドユーザの通信時間に比例して増大する傾向があった．分散ファイルサーバのブロック転送処理時間は, 書き込みの場合, ほとんど変化がないが, samba 読み込みの場合は，遅延・ロスに比例し，やや増加する傾向を示した．

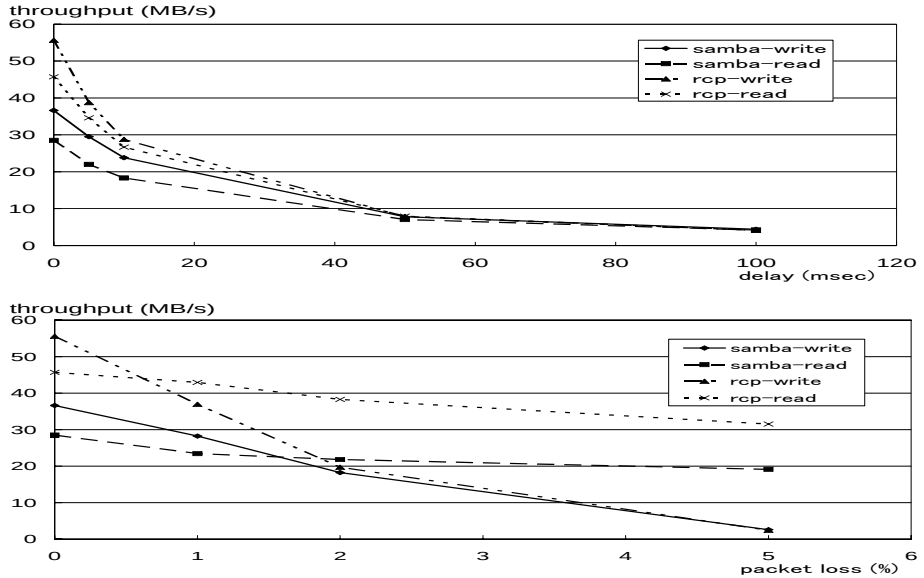

図４　システムネットワークの劣化とスループット

表４　システムネットワークの劣化とブロック転送処理時間

process time	delay	end user	client	file server
samba−write	0	2.864	1.129	0.208
	5	3.556	1.794	0.210
	10	4.420	2.672	0.218
	50	13.574	11.431	0.228
	100	23.699	21.554	0.233
samba−read	0	3.684	1.226	0.294
	5	4.781	2.340	0.781
	10	5.734	3.306	0.886
	50	14.896	12.143	1.237
	100	25.196	22.419	0.998
rcp−write	0	1.800	1.854	0.227
	5	2.582	1.794	0.212
	10	3.478	2.577	0.223
	50	12.708	11.014	0.258
	100	24.284	21.575	0.244
rcp−read	0	2.200	1.854	0.927
	5	2.894	2.467	0.903
	10	3.758	3.291	0.927
	50	12.590	11.732	0.861
	100	24.209	22.428	0.983

unit: delay (msec), end user, client, file server (msec)

process time	packet loss	end user	client	file server
samba−write	0	2.864	1.129	0.208
	1	3.760	2.000	0.205
	2	5.809	4.063	0.221
	5	41.668	40.113	0.232
samba−read	0	3.684	1.226	0.294
	1	4.479	1.986	0.895
	2	4.822	2.338	0.911
	5	5.528	3.095	0.892
rcp−write	0	1.800	1.145	0.227
	1	2.773	2.024	0.206
	2	5.159	4.306	0.231
	5	42.410	41.493	0.234
rcp−read	0	2.200	1.854	0.927
	1	2.355	1.970	0.901
	2	2.649	2.243	0.842
	5	3.325	2.860	0.851

unit: packet loss (%), end user, client, file server (msec)

4.5 プロキシサーバの過負荷の影響

次に，プロキシサーバの負荷が与える影響について評価した．プロキシサーバの負荷を増大させるため，エンドユーザ PC２～４において，エンドユーザ PC１と同じ測定スクリプトを実行した． エンドユーザ PC２～４は，負荷発生が目的のため，測定トラヒックの３倍以上の負荷を与える際には同時に複数の測定スクリプトを起動させた．負荷を増減させることにより，全トラヒックに占める負荷トラヒックの比率で計算される負荷率を 0～96.8%の範囲で変化させた．なお，本測定は samba の場合のみ実施している．負荷率とスループットの関係を図５に示す．逆転現象はあったものの，負荷率の増大に伴いスループットが低下する傾向が見られた．

次に，エンドユーザの通信時間，クライアントのブロック転送処理時間，分散ファイルサーバのブロック転送処理時間の比較結果を表５に示す．読み込み時において，負荷率の増大に伴い，クライアントのブロック転送処理時間がやや増大する傾向があった．

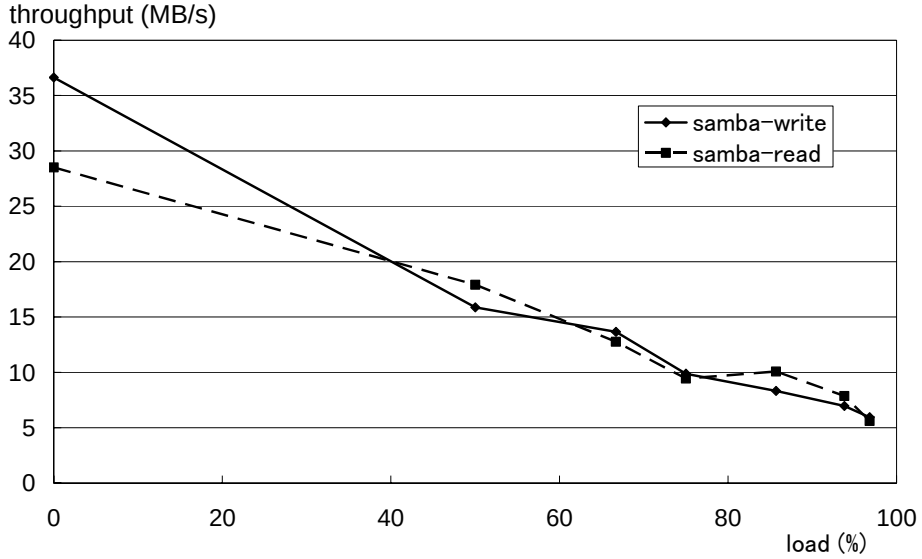

図５　プロキシサーバの負荷率とスループット

表５　プロキシサーバの負荷率とブロック転送処理時間

process time	load	end user	client	file server
samba-write	0	2.864	1.129	0.208
	50	6.665	1.140	0.213
	66.7	7.745	1.147	0.222
	75	10.683	1.173	0.244
	85.7	12.631	1.161	0.234
	93.8	15.165	1.166	0.233
	96.8	17.723	1.165	0.223
samba-read	0	3.684	1.226	0.294
	50	5.942	1.846	0.913
	66.7	8.228	1.931	0.991
	75	11.123	2.080	1.136
	85.7	10.423	2.086	1.143
	93.8	13.370	1.965	1.021
	96.8	18.777	2.150	1.206

unit: load (%), end user, client, file server (msec)

4.6　分散ファイルサーバの過負荷の影響

最後に，分散ファイルサーバの負荷が与える影響について評価した．ネットワークやプロキシサーバに影響を与えることなく分散ファイルサーバの負荷を増大させるため，分散ファイルサーバ１～４上で，測定スクリプトと同じスクリプトを用いて，負荷トラヒックとなるファイルの書き込み・読み込みを実行した．分散ファイルサーバ上の測定スクリプト実行数を制御することで，負荷率を 0～93.8%の範囲で変化させた．本測定についても samba の場合のみ実施している．負荷率とスループットの関係を図６に示す．スループットは，負荷率の増加に対して，プロキシサーバの負荷の場合とほぼ同等の割合で低下した．

次に，エンドユーザの通信時間，クライアントのブロック転送処理時間，分散ファイルサーバのブロック転送処理時間の比較結果を表６に示す．プロキシサーバの負荷の場合と異なり，負荷の増大に伴い，分散ファイルサーバのブロック転送処理時間が増大する傾向が見られた．

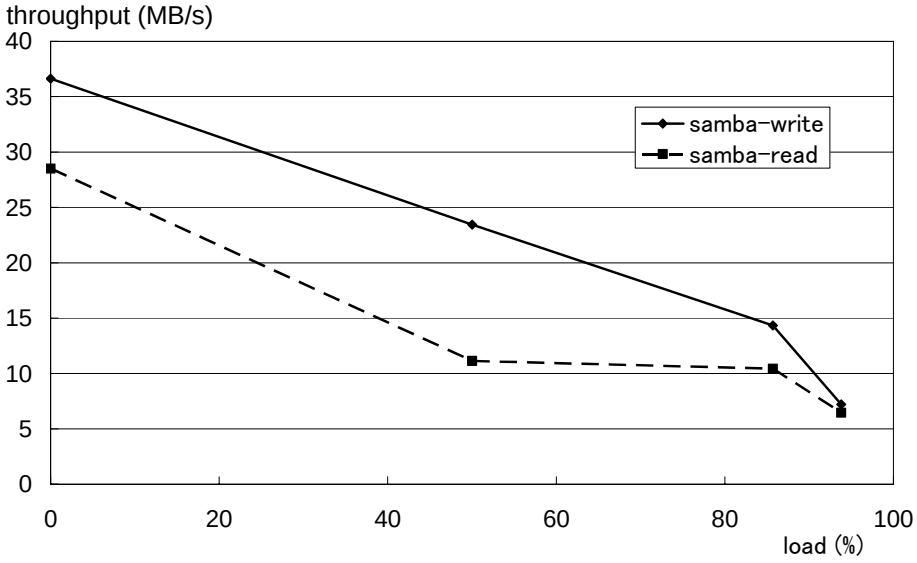

図６　分散ファイルサーバの負荷率とスループット

表６　分散ファイルサーバの負荷率とブロック転送処理時間

process time	load	end user	client	file server
samba-write	0	2.864	1.129	0.208
	50	4.734	3.035	2.082
	85.7	7.918	6.197	5.265
	93.8	16.470	14.761	13.952
samba-read	0	3.684	1.226	0.294
	50	11.343	8.913	7.982
	85.7	16.869	14.419	13.487
	93.8	33.829	31.069	30.136

unit: load (%), end user, client, file server (msec)

5. 考察

本章では，実験結果の考察と，ボトルネックの自動分析の運用監視ツールとしての適用可能性について述べる．

5.1 ボトルネック箇所の分析

まずボトルネック箇所毎に，ネットワーク・サーバの性能劣化が，クライアント，分散ファイルサーバのブロック転送処理時間に与える影響について考察する．ブロック転送処理時間が，エンドユーザの通信時間に大きく影響している場合には“劣化”，それ以外は“ほぼ影響なし”とした．実際には，ボトルネック生成時に，エンドユーザの通信時間が大きく増加した samba-read の場合，クライアント，分散ファイルサーバにおいて，わずかに性能の劣化が見られた．表７より，以下のことが考察される．

- システムネットワーク，分散ファイルサーバがボトルネックである場合には，クライアント，分散ファイルサーバの劣化度合いを観測することで，ボトルネックの特定が可能である．
- ユーザネットワーク，プロキシサーバがボトルネックの場合は，いずれもクライアント，分散ファイルサーバの性能にほとんど影響が無かったが，プロキシサーバの方が，samba-read において，やや劣化度合いが大きい傾向があった．

また，表３，４より，ユーザネットワーク，システムネットワークの劣化が，遅延，パケットロスのいずれであっても，クライアントや分散ファイルサーバへの影響の度合いが変わらないことが確認できた．

表７　ボトルネック箇所毎のサーバへの影響

ボトルネック	クライアント	分散ファイルサーバ
ユーザネットワーク	ほぼ影響なし*1	ほぼ影響なし*1
システムネットワーク	劣化	ほぼ影響なし*1
プロキシサーバ	ほぼ影響なし*1*2	ほぼ影響なし*1
分散ファイルサーバ	劣化	劣化

注*1: samba-read の場合のみ僅かに劣化した．
注*2: samba-read において，ユーザネットワークの場合に比べて，やや劣化の度合いが大きい傾向があった．

5.2 異常値

次に，個々の測定における異常値について考察する．プロキシサーバ，分散ファイルサーバは，本測定専用に使用しており，外的な影響が無いことを前提としているものの，OS の動作などにより，一時的にブロック転送の処理ができず異常値が発生する可能性がある．また，４章での測定値は，10 回の測定値の平均を用いた結果を示しているが，平均値として異常があった場合に，個々の値の散らばりについて分析する必要がある．そこで，ボトルネックなし，ユーザネットワーク・システムネットワークの遅延 100msec，プロキシサーバ，分散ファイルサーバの負荷率を 96.8%，93.8%とした場合に関し, samba で 100 回の書き込みの測定を行った. その結果を表８に示す．

プロキシサーバに負荷を与えたとき(proxy server load 96.8%)のプロキシサーバ，分散ファイルサーバのブロック転送処理時間は，平均値として正常時とほぼ同じ値であるものの，最大値が正常値の２倍近くの値を示した．99%値については，平均値に近い値を示していることから，偶発的な異常値の可能性があると考えられる．

分散ファイルサーバに負荷を与えたとき(file server load 93.8%)のプロキシサーバ，分散ファイルサーバのブロック転送処理時間は，他と比較して大きな標準偏差値を示した．一方，95%値や 99%値も最大値に近い値であるため，少数の異常値によるばらつきではないことが分かる．

表８　100 回測定時の統計値

proxy server processing time

statistics	normal	user delay 100msec	system delay 100msec	proxy server load 96.8%	file server load 93.8%
mean	1.137	1.169	21.610	1.180	13.071
min	1.105	1.125	21.244	1.127	1.946
max	1.209	1.269	25.471	2.464	23.929
99percentile	1.176	1.260	22.772	1.430	22.919
95percentile	1.187	1.226	21.671	1.249	21.667
stdev	0.02	0.027	0.412	0.138	4.468

file server processing time

statistics	normal	user delay 100msec	system delay 100msec	proxy server load 96.8%	file server load 93.8%
mean	0.219	0.228	0.219	0.237	12.226
min	0.187	0.185	0.187	0.193	0.996
max	0.291	0.34	0.291	1.526	23
99percentile	0.259	0.328	0.02	0.494	22.444
95percentile	0.268	0.285	0.268	0.31	20.746
stdev	0.02	0.028	0.259	0.139	4.463

5.3 運用監視ツールとしての適用

多数のサーバを監視するツールとして，オープンソースの Nagios[6]や Zabbix[7]が広く利用されている．これらは，CPU 負荷，ディスク使用量などサーバリソースの状態や，HTTP, FTP などのネットワークサービスなどを監視する．また，文献[8]は，複数のサーバのリソース状態をグループで監視することにより，サーバシステムとしての健全度を分析する手法を提案している．しかしながら，CPU 負荷など，OS レベルでの異常値が，必ずしも分散ファイルシステムの性能劣化と関連するとは限らない．

これに対して，筆者らの手法は，分散ファイルシステム内で，性能に影響する通信動作に関する通信ログを収集するため，実験結果で示したように，エンドユーザの品質の低下の要因となるボトルネック箇所を特定することが可能である．実験においては，偶発的と想定される異常値が発生する場合も存在したが，多くのサンプル値を収集できる実運用においては，異常値を除去することで，より正確に異常判定を行うことが可能となる．

6. おわりに

本稿では，ネットワークの遅延・パケットロスやサーバの過負荷が，分散ファイルシステムの性能に与える影響を評価するとともに，分散ファイルシステムのボトルネック箇所の特定手法について述べた．今後の課題として，通信ログの収集および分析処理を自動化する運用監視ツールの開発，異常と判定する閾値の設定，メタデータサーバも含めたボトルネック特定手法の検討などが考えられる．

参考文献

1) Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Carlos Maltzahn, “Ceph: A Scalable, High-Performance Distributed File System.” OSDI 2006, November 2006.
2) Gluster, Inc., “Gluster.org Community Website,” http://www.gluster.org/, 2011.
3) S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google file system. In Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP ’03), Bolton Landing, NY, Oct. 2003.
4) Osamu Tatebe, Kohei Hiraga and Noriyuki Soda, “Gfarm Grid File System,” New Generation Computing, Ohmsha, Ltd. and Springer, Vol.28, No.3, pp.257-275, DOI: 10.1007/s00354-009-0089-5, 2010.
5) FUSE: Filesystem in Userspace, http://fuse.sourceforge.net/.
6) Nagios Enterprises, LLC. “Nagios – The Industry Standard in IT Infrastructure Monitoring,” http://www.nagios.org/
7) Zabbix SIA, Homepage of Zabbix:: An Enterprise-Class Open Source Distributed Monitoring Solution, http://www.zabbix.com/
8) 寺下雅人，西谷明彦，大岸智彦，“分散サーバシステムの自律的な運用方法の検討，” L-015, FIT 2010, September 2010.