モバイルコンピューティングと
ワイヤレス通信　23－9
高度交通システム　11－9
（２００２．１１．２８）

ActiveBB/ResponseServer:
サービス適応による大規模アクセス応答サーバ構築支援システムの試作

落合 勝博　　大芝 崇　　田淵 仁浩　　小池 雄一　　神場 知成
NECインターネットシステム研究所

本稿では、サービス適応による応答処理技術の提案と、その試作および評価について述べる。双方向TV のように潜在的に数千万規模のユーザを抱えるサービスでは、ユーザレスポンスが短時間に応答サーバに集中する問題がある。従来のサーバ構築技術では、この問題を解決できなかったり、できたとしても実サービスに導入するにはコストがかかりすぎたりする。サービス適応では、アプリケーション知識に基づいた高速処理可能なモデルと、実サービス上の制約に基づいた実用解を導く数理モデルによって、大規模なユーザレスポンスに対応可能なサーバを実現する。

Development of Service-adaptable Response Server : ActiveBB/ResponseServer

Katsuhiro Ochiai, Takashi Oshiba, Masahiro Tabuchi, Yuichi Koike and Tomonari Kamba
NEC Corporation, Internet Systems Research Laboratories

This paper describes the response-processing method based on service-adaptation, its development and valuation. Interactive TV service has a problem of concentrated-responses. Conventional server-construction techniques are not effective for concentrated-responses. Service-adaptation method realizes the effective server for such responses with both high-speed processing models based on application-knowledge and calculation models based on actual restrictions.

1. はじめに

BS デジタル放送を始め、CS110°デジタル放送、地上波デジタル放送では、双方向性を持つ番組制作が可能となっている。放送サービスは、多くの人に同時に情報配信することができる点で優れている。しかし、双方向性を持ったことで、短時間にユーザレスポンスが集中する問題(バーストレスポンス)を抱えており、大規模アクセスに対応可能な応答サーバの必要性が高まっている。本稿では、このような大規模アクセス対応の課題と、その解決を図る試みである「サービス適応」の考え方、また、双方向 TV サービス向け応答サーバの試作について述べる。

2. バーストレスポンス対応の必要性と課題

2.1. 双方向 TV サービスにおけるバーストレスポンス対応の必要性

既に放送が開始された BS デジタル放送、CS110°デジタル放送、および2003年から3大広域圏で開始予定の地上波デジタル放送では、データ放送[1]が可能となっている。データ放送では、BMLと呼ぶHTMLに類似するフォーマットで記述されたコンテンツをユーザ端末に送信し、そのユーザレスポンスを Java スクリプトを使って通信経由で局側サーバに返信できる。この仕組みを使った双方向性を備えたTVサービスは、今後10～15年程度

で普及することが予想され、視聴者が参加可能な新しい形の番組の普及が期待されている。

しかし一方で、放送が持つ同報性故に、双方向 TV サービスでは、ユーザレスポンスが特定時間に集中する問題がおこる。現在の BS デジタル端末普及台数は 150 万台弱(2002 年 8 月末時点、JEITA (株)電子情報技術産業協会 調べによる)であるが、将来 2010～2015 年頃には日本の推定世帯数 5000 万弱(国立社会保障・人口問題研究所による)に近い数の端末が普及する可能性がある。したがって、サービスの内容と視聴率によっては、将来、一千万規模のバーストレスポンスを、局側サーバで捌く必要が出てくる。

2.2. 従来技術による対応方法

このような大規模ユーザレスポンスに対処するためには、以下の 2 つの方法がある。

(a) サーバシステムの性能を上げる方法

一つには、ハードウェアをより高速なものに置換したり[2]、ソフトウェア処理しているもののうち特に遅い部分をハードウェア化する方法がある。

もう一つは、負荷分散による性能改善がある。キューを使ってパイプライン処理をしたり、ロードバランサを使ったクラスタリングを行うことで並列動作による処理の負荷分散を行う。

(b) レスポンスを集中させない方法

現在BSデジタル放送のある番組で取られている方法では、参加者が一定条件を満たしているか端末側で判断し、条件を満たした端末だけが局サーバにアクセスする。

また、端末側で遅延発呼して、アクセスが集中しないよう時間方向に処理を分散している。

NTT の TeleCollection[3]や Joinet[4][5]では、アクセスの許可条件を放送することで、端末からのアクセス数をコントロールする。

2.3. 従来技術の課題

前節(a)で述べたサーバシステムの性能を上げる方法は、既に Web サイト構築で一般に行われている。それにも関わらず前節(b)で述べたように実際の TV サービスではアプリケーションに特化してレスポンスを集中させない工夫を行っている。これは、現在 Web サイト構築一般に利用されている技術では、双方向 TV サービスのようなバーストレスポンスへの対処に一定の限界があるからだと思われる。原因の一つは、TV の特徴の一つであるライブ性を、大規模なユーザアクセスがある時に実現しづらいことにある。もう一つは、大規模な処理を実現する多数のサーバを導入・運用するコストが高いことにある。

(i) ライブ性の限界

例えば、参加者の中からポイント上位者を抽出するといった処理を、一般的な設計で実装すると、図 1 のようになる。この例では、ロードバランサによって処理を Web/AP サーバに振り分け、各 Web/AP サーバは受け付けたポイント情報を逐次 DB サーバに格納する。参加ユーザ数の増加には、Web/AP サーバを追加することで対応する。しかし、DB サーバの部分は最後の抽出処理のためにデータを一箇所に集める必要があり並列化できないので、一定の台数まで Web/AP サーバを増やすとそれ以上増やしても性能の向上は望めない。したがって、このボトルネックが顕在化するほどの大規模なユーザアクセスが発生する場合には応答のライブ性は失われる。

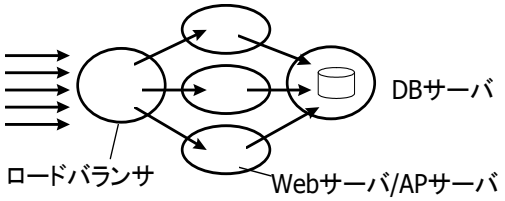

図 1: 汎用技術によるライブ性の限界

(ii) 導入・運用コストの高さ

前記のような一般的な方法で、上位者抽出を行うには、例えば抽出処理の部分のライブ性を放棄して DB サーバを並列化する方法が考えられる(図 2)。この場合、ポイント回収のライブ性は保証されるので、次回番組冒頭で上位者を発表するというようなことは可能となる。しかし、同じユーザ数で図 1 の構成と比較した場合、DB サーバの増加分だけ余計にマシンが必要となっている。したがって、導入、運用にかかるコストが余計に必要となる。また、このように設計されたシステムはこのサービス専用にしか使えない。実サービスでは、特に初期には採算が取れない場合が多いので、この大規模な専用システムを持つコストはサービス自体の継続に大変な負担となる。

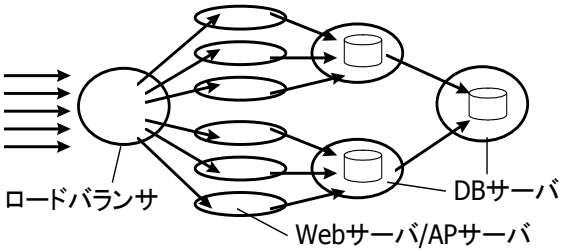

図 2: 抽出のライブ性を放棄した実装方法

一方、前節(b)で述べたような方法は、参加ユーザ数が増えても応答サーバの能力を増強する必要がないというメリットがある。しかし、全てのユーザからのレスポンスを受けるには、ライブ性が失われるという欠点もある。例えば TV ショッピングの申し込み、またはアンケートで意見を集約するといったようなことには不向きである。

3. サービス適応による応答処理

前章で述べた問題を解決するために、我々は、「サービス適応」による応答処理技術を提案している[6]。サービス適応による応答処理では、アプリケーション知識に基づいた①大規模アクセスに対応かつ高速な処理モデルの設計によって、大規模な応答処理に対してもライブ性を確保し、②実サービスにおける制約下での実用処理の決定と運用によって、導入・運用コストを低減する。

① 大規模アクセスに対応かつ高速な処理モデルの設計

ボトルネックを分散化するシステム構成をサービス分類毎にモデル化する。このモデルでは、処理サーバ台数の単純な増加によって全体性能を改善しやすい処理構成とする。

※サービス分類とは、ユーザ見えが若干異なっても基本となるアーキテクチャが同一のものは一つのサービスとして扱う分類。例えば、「アンケート回答回収」と「プレゼント応募受付」は、機能面で見ると同じ「データ回収保存」というサービス分類とみなせる。

② 実サービスにおける制約下での実用処理の決定と運用

まず、あらかじめ①で用意した各モデル毎に、各種の変動パラメータ(想定ユーザ数、使用可能なマシン台数上限、受付時間、集計時間など)による性能への影響を数理モデルで表現する。サーバ運用者は、運用時に処理モデルの選択を行い、またサービスを行う上での制約条件を与える。すると、先ほどの数理モデルを使って制約条件を満たす運用が可能なシステム構成を演算する。応答サーバでは、この演算結果にしたがった構成と処理内容でサービスを処理する。

3.1. 大規模アクセスに対応かつ高速な処理モデルの設計

我々は、双方向 TV サービスにおける番組特性を分析し、その結果、サーバの実時間処理性能を元に双方向 TV 番組を 2 つのグループに分類した。

A. 番組時間中にユーザレスポンスを回収し、その結果を番組中に活かすグループ
音楽番組における曲のリクエストや、視聴者参加型のクイズ番組、選挙での投票番組等

B. 番組時間中にレスポンスは集中するが、結果を番組中に活かす必要のないグループ
ユーザ登録、ショッピング等

A には、広い意味では番組補完情報提供(あらすじ提供、地域毎の天気予報等)が含まれるが、読み込み主体の応答のため、キャッシュや CDN 等の既存技術[7]によって対応可能と考え、今回のモデル化対象から外した。

次に、上記分類を元に、グループ A に対応する高速集計モデルと、グループ B に対応する高速回収モデルを設計した。これらのモデルでは、フロントエンドに位置する回収サーバの台数を増加させると、単位時間あたりの受付可能数がそれに比例して増加するようにしてある。つまり、サーバ内部での特定箇所のボトルネックが原因となるスケーラビリティの上限は理論上存在しない設計となっている。

① 高速集計モデル

図 3 に、ユーザからの回答を放送中の限られた時間内に回収し、その回収データから得られる集計結果を番組中に利用するためのモデルを示す。このモデルは、回答の集計／抽出結果だけが重要な番組に向く。例えば投票(投票割合や投票率の集計)、スコアランキング(スコア上位者のみをソートで抽出)、プレゼント応募(当選者情報のみを抽出)、オークション(最高額をつけた人の抽出)等である。

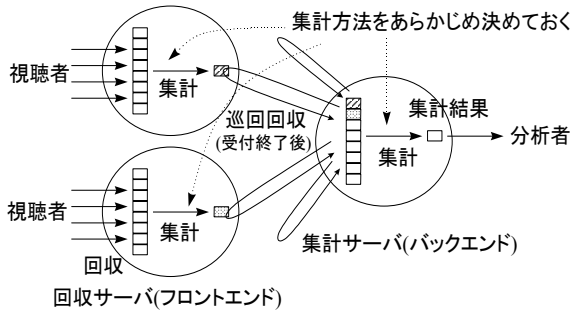

図 3:高速集計モデルの概念図

2 章で説明したように、従来技術では、回収時に集計サーバ(2 章の課題説明では DB サーバに相当)で全ての回答を集めると、集計サーバへのアクセスがボトルネックとなり、回収サーバを増やせない。そこで、並列配置される回収サーバで回答を回収し、対象項目の一次集計を各サーバで行う。そして受付時間の終了後、各回収サーバの集計結果を集計サーバに集めて全体での二次集計を行なう。こうして、一次集計によって集計サーバに送信されるデータ量と頻度が削減されるので、多数の回収サーバを並列化しても巡回回収がボトルネックとならない。また、一次集計によって、集計サーバへのデータ送信量を各回収サーバで処理したユーザ数と独立にする。

また、一次集計、二次集計とも実行時にリアルタイムで実行され、この時の計算は、あらかじめ決められた集計方法に従うので、集計方法は、実行前に計画しておく必要がある。この集計方法については、先の番組分類から①合計、②単純集計、③クロス集計、④抽選、⑤先着者抽出、⑥上位／下位者抽出を、必須機能として抽出した。各集計方法は、このモデルに基づいて回収サーバ、集計サーバにおいて実行されるが、サーバ内部での処理自体は集計方法毎に異なる。

② 高速回収モデル

図 4 に、ユーザからの回答を放送中の限られた時間中に回収して回収データを自由に分析利用するためのモデルを示す。このモデルは、ユーザの回答を全て保存したいが、応募時間は限定される番組に向く。例えば新譜 CD の予約(TVCM で宣伝後、BS デジタル端末で応募受付する等を想定)や TV ショッピング(TV 中で宣伝と応募受付を同時に行なうことを想定)等が挙げられる。

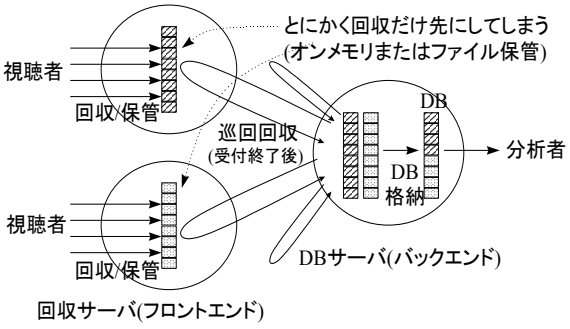

図 4: 高速回収モデルの概念図

2 章で説明したように、従来技術では、回収時に中段にある DB サーバで一次格納処理を行なうと、余計なサーバ台数が必要となりコスト増となる。そこで、並列配置される回収サーバで回答を受け付けると、まず内部で高速な媒体に一次格納する。そして受付時間の終了後、各回収サーバで格納したデータを DB サーバに転送し、全てのデータを転送した後で DB 上に格納しなおす。このような 2 段階のパイプライン処理によって応答処理のスループットを上げられる。その結果、回収サーバを増やしても DBサーバへのアクセスが回収時にボトルネックとならない。また、受付終了後に一旦 DB サーバに対して DB へのアクセスを伴わないデータ転送を行なうことで、時間のかかる DB アクセス完了を待たずに回収サーバを自由にすることができるので、回収サーバを再び別の仕事に割り振ることができ、システム全体の稼動性が高まる。

3.2. 実サービスにおける制約下での実用処理の決定と運用

実運用を無視して、前節のモデルを使った最速な構成を取るとすれば、式 1 に示すような数理モデルを組み立て、処理時間総和 f()が最小となる回収サーバ台数 M を求めればよい。

$$f(Uf,M)=Pt(Uf)+Ps(Uf)+N(M)+Pb(M)$$

ユーザ数/台　回収サーバ台数

処理時間総和　回収時間関数　一次集計時間関数　巡回回収時間関数　二次集計時間関数

式 1: 高速集計モデルの処理時間等式

例えば、Pt(1)=Ps(1)=N(1)=Pb(1)=1(秒)であり、総ユーザ数を 20 とした場合の処理時間総和 f()の値の推移を図 5 に示す。この場合には、回収サーバ台数 M=4 の場合に処理時間総和 f(20,4)=18 が最小値となる(ただしここでは問題を簡単にするため、サーバ処理開始時において全てのリクエストが待機中であると仮定)。

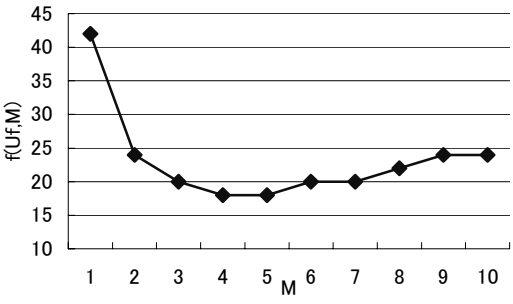

図 5: 高速集計モデルによる最速解

しかし、例えば導入コストの問題から回収サーバ台数は 2 台が限界だとすれば、処理にかかってよい時間を 24 秒まで緩和すればサービスが可能となる。また、逆に処理は 20 秒かかっても構わないという条件が与えられるならば、このサービスの運用には 3 台の回収サーバがあればよい。このように、実サービスでは、制約を考慮することで、コストを抑えた実用的な運用が可能となる。

そこで我々は、実サービスにおける制約下での実用処理の決定のために、制約を含む各種のパラメータの不等式として数理モデルを設計した。

式 2 は、高速集計モデルの数理モデルである。この数理モデルでは、回収サーバでの処理時間関係式と集計サーバでの処理時間関係式を同時に満たす制約条件値の組み合わせを求めると実用解を求めることができる。不等式内の各評価関数は、このモデル上での集計処理によって異なる。例えば、単純集計と、抽選ではプログラム上では異なる計算量やデータ量となるので、それぞれ別の評価関数が使われる。

式 3 は、高速回収モデルの数理モデルである。この数理モデルでは、回収サーバでの処理時間関係式と DB サーバでの処理時間関係式を同時に満たす制約条件値の組み合わせを求めると実用解を求めることができる。なお、高速集計モデルと異なり、評価関数は固定である。

回収サーバでの処理時間関係式

$$Pt(Uf) + \sum_{k=1}^{Lr} Ps_k(Uf) \le Rr$$

回収時間関数　ユーザ数　集計時間関数1　回収時間　集計項目数

集計サーバでの処理時間関係式

$$Nc(M) + \sum_{k=1}^{Lr} Nt_k(M) + \sum_{k=1}^{Lr} Pb_k(M) \le Rs$$

ネットワーク接続時間関数　データ転送時間関数　回収サーバの台数　集計時間関数2　集計時間　集計項目数

式 2: 高速集計モデルの数理モデル

回収サーバでの処理時間関係式

$$Pt(Uf) + \sum_{k=1}^{Lr} Pm_k(Uf) \le Rr$$

回収時間関数　ユーザ数　格納時間関数1　回収時間　回収項目数

DBサーバでの処理時間関係式

$$Nc(M) + Nt(Uf, M) + Pw(Uf, M) \le Rs$$

ネットワーク接続時間関数　データ転送時間関数　回収サーバの台数　格納時間関数2　撤退までの猶予時間　ユーザ数

式 3: 高速回収モデルの数理モデル

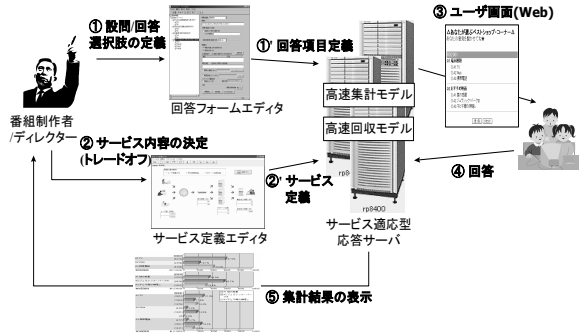

図 6: 試作したシステムの構成と利用手順

4. サービス適応型応答サーバの試作

サービス適応による応答処理を行うサービス適応型応答サーバ"ActiveBB/ResponseServer"と、応答サーバの構築・運用支援を行うツール群を試作した(図 6)。サーバの実装には、OS: Linux (Redhat 系)、Web サーバ: Apache + Tomcat、DB: PostgreSQL、アプリケーション: Java(本体: Servlet + UI: Applet)を用いた。

この試作システムは、回答フォームエディタ、サービス定義エディタ、サービス適応型応答サーバから構成される。

A. 回答フォームエディタ

エンドユーザに対する設問項目の設計を行う(図 7, ActiveCR[8]のアンケートエディタを流用した)。この結果は、エンドユーザに示される回答画面、サービス定義エディタでの処理対象指定、エンドユーザからサーバに寄せられる回答の受付 I/F として利用される。

図 7: 回答フォームエディタ画面

B. サービス定義エディタ

回答フォームエディタで設計した各設問に対して、サーバでどのような処理を行うかを定義する。まず、集計方法や回収方法等を UI で入力する(図 8)。次にサービスの実行制約を入力する(図 9)。すると、式 2 式 3 等を用いて実用解を提示する。応答サーバでは、この定義に従ったサービス適応処理を行う。また、この際に、もし式を満たす解がなければ、制約値の改善候補を表示する。

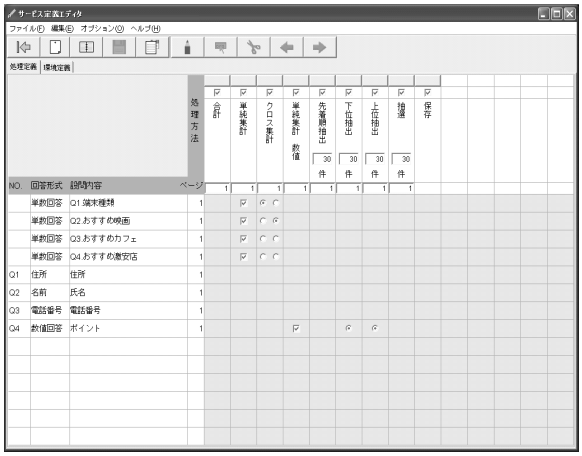

図 8: 処理内容定義画面例

図 9:実行環境定義画面と検証結果例

C. サービス適応型応答サーバ

エンドユーザからの回答を受けつけ、その回答をサービス適応処理する部位と、そのための運用管理を行う部位とから構成される。サービス適応に従ったサーバ処理については、サービス定義エディタの定義結果に従い、前述のモデルで説明したように動作する。また、運用管理には利便性を考え以下の機能を実装した。

(i) サービス登録機能

回答フォームエディタとサービス定義エディタによる定義結果をサービスとして登録し、実行スケジュールを予約する。

(ii) ユーザ回答用 Web 画面生成機能

回答フォームエディタの定義(XML)から UI(HTML)を自動生成する(図 10)。

(iii) スケジュール機能

サービス登録で行ったスケジュールに従って、フロントエンドサーバ、バックエンドサーバに必要な定義ファイルの配布と起動、終了処理を行う。

(iv) レポート機能

サービス適応によって得られた集計結果を UI として表示(図 11)もしくはテキストファイルとして出力する。

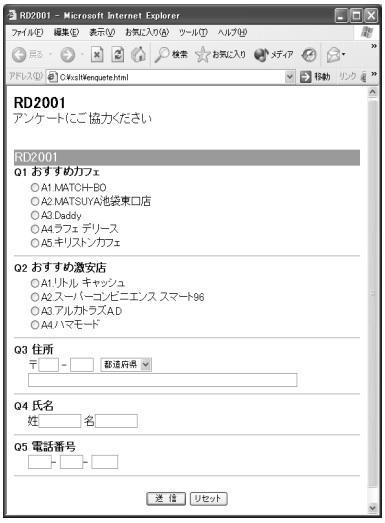

図 10: 自動生成されたユーザ回答用 Web 画面例

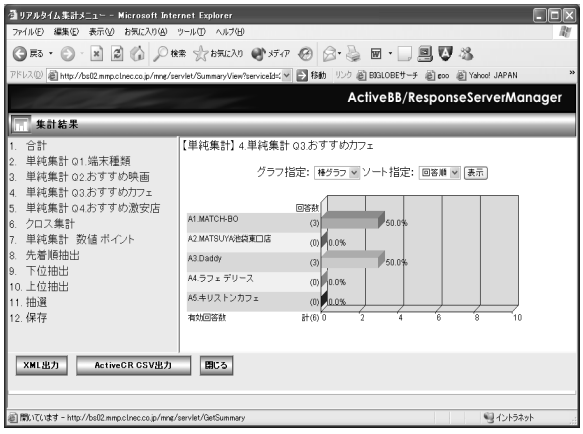

図 11: 集計結果画面例(UI)

5. 評価

図 12 は、回収サーバ台数の上限値を制約として与えた場合に、サービス定義エディタによる処理時間算出結果がどのように推移するかを表している。この評価は複数台の Pentium4-1.2GHz の PC で一千万人から選択式回答を単純集計する場合のものである。

受付時間のグラフが示すように、千台規模になっても反比例して受付時間が短縮されており、このことから十分なスケーラビリティのあるモデル設計であることが確認された。また、約 800 台使用した場合には 3 分弱で総処理(=受付から集計まで)を完了している。受付だけは 1 分という制約がある場合でも、約 1100 台の回収サーバを並べることで受付を終了し、かつその後 2 分弱で集計を完了する。このことから、このモデルは、放送番組中で処理結果をフィードバックできる十分なライブ性を持つことも確認された。またこのような算出結果を示せること自体によって、コスト等の制約を元にした実用解を算出可能なことが確認された。

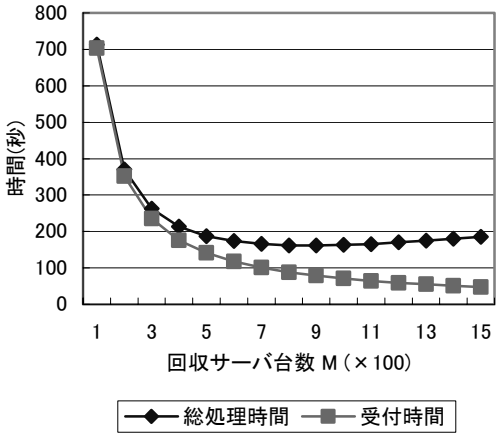

図 12: 単純集計の最速解

6. まとめと課題

本稿では、サービス適応による応答処理技術と、放送向けの試作について報告した。

サービス適応による応答処理では、アプリケーション知識を使ったサービス種類毎の高速処理モデルと、制約を加えた数理モデルによる演算によって、高速性と実用的なコストを兼ね備えたシステム構成を可能とする。今後は以下の課題について研究を行う予定である。

(1) サービス定義エディタのユーザビリティ評価とシミュレーション結果の正確性
(2) 実環境を用いたスケーラビリティの検証
(3) 予想を上回るユーザレスポンスへの対応
(4) 他サービスへの展開と汎用化

7. 参考文献

[1] デジタル放送におけるデータ放送符号化方式と伝送方式,ARIB STD-B24,電波産業会

[2] 「DB システムの高速化に新手法」, 日経インターネットテクノロジー2002 年 3 月号, pp.93-101, 2002

[3] 渡部、岸田, 「インタラクティブ番組におけるデータ集約制御手法」, 情報処理学会マルチメディア通信と分散処理研究会、DPS-107, pp. 61-66, 2002

[4] 岸田他, 「放送・通信パスの結合による多数エンドノードからのランキング高速集計手法」, 電子情報通信学会 1997 年通信ソサイエティ大会プログラム, p.171, 1997

[5] 岸田他, 「放送・通信パスの結合による参加型クイズ番組の早押し判定法」, 情報処理学会第 55 回全国大会(3), pp.753-754, 1997

[6] 落合他, 「バーストレスポンスに耐える ASP サーバを実現するサービス適応型応答サーバ」, 情報処理学会第 64 回全国大会(3), pp.499～500, 2002

[7] 田代他、「インターネットコンテンツ配信技術の最新動向」、情報処理 Vol.42 No.11 、pp.1082-1091

[8] http://www.activecr.com/