「*Internet*広域分散協調サーチロボットの研究開発」
研究成果報告書

村岡洋一
E-mail: muraoka@muraoka.info.waseda.ac.jp
学校法人　早稲田大学

概　要

WWW サーバのデータを自動的に収集するプログラムを一般に WWW ロボットと呼ぶ．この WWW ロボットが全 WWW サーバにアクセスし，必要な WWW データを収集するには，現状技術では，WWW ロボットを動作させるサーバの性能および回線容量の制限などから，日本国内のみの WWW サーバを対象とした場合でも１ヶ月以上の時間を要する．

このような現状を打破するため，本研究開発では，広域な範囲の WWW データをインターネット上に分散された複数の WWW ロボットで協調して高速収集するための，分散協調サーチロボット（ソフトウェア）の研究開発を行い，日本国内 (ドメイン名が jp で終わる WWW サーバ) の全 WWW データを 24 時間以内に収集することを目標とする．

初年度である平成 10 年度は，「分散協調サーチロボットのプロトコルの検討」，「分散エリア決定方法の検討」，「分散協調サーチロボットプロトタイプシステムの設計と先行評価」を行った．

実際に，早稲田大学，京都大学，北陸先端大学院大学，慶應義塾大学，大阪府立大学，シャープ (株)，電子技術総合研究所の 7 個所に分散されたプロトタイプシステムを用い，103 の WWW サーバを対象とした収集実験を行い，実際の分散協調サーチロボットの有効性を確認した．具体的には，一カ所で集中して収集する場合に比較し，ランダムな分散をした場合，2.6 〜 10.6 倍の高速化が可能であり，負荷均一化を行った場合，5.5 〜 22 倍の高速化が可能であることがわかった．

今後は，「分散協調サーチロボットの開発」，「分散協調サーチロボット環境の実装」，「分散協調サーチロボットの評価」の 3 点を日本国内に設置した 30 台以上のサーチロボットによって行い，jp ドメインに存在する全 WWW サーバのデータを 24 時間以内に収集するシステムを構築し，運用を開始する予定である．

目次

1 はじめに

World Wide Web (WWW) は，Mosaic が開発されて以来，インターネット上での情報提供及び情報収集の手段として世界中で利用されている．しかし，WWW 自体は，複数の WWW サーバから特定の情報を検索するプロトコルを持たないため，WWW 上のデータを対象とした検索を行うためには，既存の WWW アーキテクチャ上で検索システムを構築しなければならない．このようなシステムの代表例として，AltaVista [1] や HotBot [2] 等が挙げられる．これらは，HTTP クライアントの一種である WWW ロボット [3] と呼ばれるソフトウェアを用いて WWW サーバ上のデータを収集し，収集したデータに対する検索を提供している．WWW ロボットは，次々と WWW サーバにアクセスし，HTML ファイルを自動的に取得する．そして，収集した HTML ファイル内のリンクを辿り，別の WWW ページを取得するという一連の動作を繰り返すことにより，WWW 上のデータを収集している．

現在の WWW ロボットは，AltaVista や HotBot をはじめとする各々の検索システムが個別に動かしており，世界中に 192 の WWW ロボット (1999 年 2 月現在) が存在する [4]．そして，一つの WWW サーバに対して数十の WWW ロボットが訪れているのが現状である．このような状況は，ネットワークトラフィック及び WWW サーバに与える負荷の増大をもたらしている．また，WWW サーバ数は，指数関数的に増大しており (1999 年 2 月現在約 430 万台 [5])，WWW ロボットでの収集時間も指数関数的に増大している．

このような問題を解決する方法として，Harvest [6] が米国において研究されてきた．Harvest では，Gatherer と呼ぶ WWW ロボットをネットワーク上の複数個所に配置し分散収集をすることができる．また，Broker と呼ぶインタフェースを用いることにより，複数の Gatherer が収集したデータに対する検索を可能としている．しかし，分散収集時の WWW サーバの割り当ては，静的にしか設定することができず，分散を自動的に行うための仕組みを持っていない．

これに対して「Internet 広域分散協調サーチロボットの研究開発」では，

① WWW ロボットをネットワーク上に分散して複数配置する．

② 分散した WWW ロボットが担当するサーバを自動的に決定させると共に，協調動作させる．

の 2 点により，高速に WWW サーバ上のデータを収集することを目指す．このような分散協調型 WWW ロボット－「Internet 広域分散協調サーチロボット」の研究開発により，直接の効果・成果，及び，波及効果として以下の 4 点が期待される．

① WWW サーバ上のデータを高速収集（目標は日本全国の WWW サーバ上のデータを 24 時間以内に収集) することが可能となる．

② 高速収集により，検索サービスサイトでは，最新のデータを利用した検索サービスの提供が可能となる．つまり，最新性において質の高い検索を提供する

ことができるようになり，インターネット検索ビジネスの活発化にもつながると予想される．

③ 現在各検索サービスサイト毎に独立に動かしているWWWロボットを本研究開発によって開発された広域分散協調サーチロボットに置き換えることにより，国レベル，あるいは，世界レベルでの協調収集が可能となる．これにより，現在インターネットの負荷の70％を占めるhttpプロトコルによるデータ転送の内，WWWロボットによる負荷を大幅に削減することが可能となる．

④ 本ソフトウェアは次世代のコンピューティング環境として近年研究開発が活発化の兆しを見せている「広域分散コンピューティング」の一つの重要なアプリケーションとなり，本分野の研究開発の活性化に寄与する．

2　インターネット発展の推移とWWWロボット

1969年のアメリカ国防総省高等研究計画の出資によるARPAnet (Advanced Research Project Agency Network) に始まるインターネットの歴史は，1994年頃からのWWW(World Wide Web) ブームにより，研究者たちだけのものから，一般大衆のものへと大きな変化を遂げた.

この変化は数値で見ても明らかであり，インターネットに接続するコンピュータ台数は，1994年1月に220万台であったものが，1995年1月に585万台，1996年1月に1,435万台，1997年1月に2,182万台，1998年1月に2,967万台と増加した [7]. 1997年に一度落ちた伸び率は，1998年再び増加に転じ，1999年1月現在で，4,323万台のコンピュータがインターネットに接続されている [7]．また，インターネットに接続された組織数(ドメイン数) は，1999年1月現在で約432万組織である [7].

1999年2月現在のWWWサーバ数は全世界で約400万台 [8] と推定される．それらのサーバから提供されるHTMLページ数はNEC北米研究所のSteve Lawrence氏とC. Lee Giles氏が1998年4月3日号の米科学雑誌サイエンスに掲載した研究成果 [9] によれば(1997年12月現在)，3.2億と推定されており，この数値を元に単純計算すると，1999年2月現在のWWW数は，約6億ページとなる.

このような膨大な情報の中から，必要な情報を瞬時に，かつ，的確に見つけ出すための仕組みがWWW情報検索サービスである．現在のWWW情報検索サービスでは，世界中の3分の1～6分の1程度のページを対象として，収集，索引化を行い検索を可能としている．しかし，指数関数的に増大するデータをカバーするのは，現実問題として困難になりつつある．本節では，最近のWWW情報検索サービスの動向をサーベイするとともに，指数関数的に増大するデータに対処するための試みとして我々の研究以外のインターネット広域分散サーチロボットについて紹介する.

2.1　WWWサーバ数の推移

WWWは，欧州のCERN [10] と呼ばれる高エネルギー物理学の研究所が開発した分散データベースである．1993年に，このWWW上のデータにアクセスするためのWWWブラウザを米国のイリノイ大学の研究グループが開発し，Mosaic [11](現在のInternet Explorerの原型) という名で無料で一般に公開したのをきっかけとし，爆発的にその利用が進んでいる.

図1に示すように，WWWサーバ数は1994年以降急激に増加しており，1998年の一年間だけでも前年比で約2.2倍となった [8]．また，インターネットに接続されるホスト数に対する割合は，1999年1月現在で約9.4%であり，前年比約2.9%増となった.

2.2　WWW情報検索サービスの分類

WWW情報検索サービスは，その仕組みによって図2に示すように，大きく３つに分類される [12].

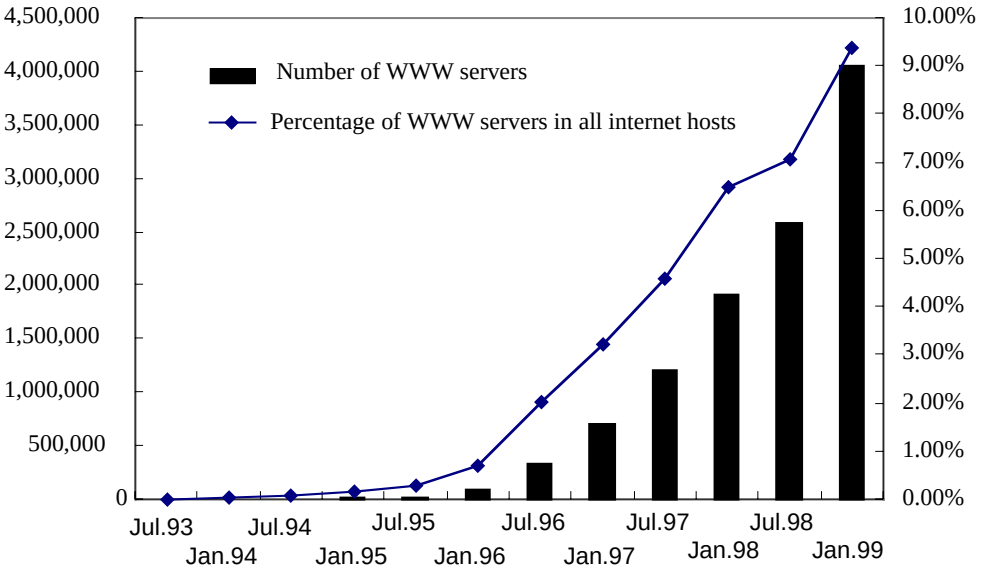


図 1: WWW サーバ台数と全ホスト数に占める割合の推移 ([7] [8] のデータに基づき作成)

ロボット系検索サービス ロボット系検索サービスでは，WWW ロボットやスパイダーと呼ばれる Web 探索プログラムを用いて，インターネット上で見つけることのできる WWW サーバー上の情報を定期的に収集し，その情報の索引付けを自動的に行っている．例として，AltaVista [1] や Goo [13] が挙げられる．これらの検索サービスでは，探したい情報に関連するキーワードを入力することによって，目的のサイト (WWW サーバ) を検索できる．ロボット系検索サービスでは，コンピュータによって自動的に全世界のデータを収集しているため，情報量が多いという利点を持つ．一方，各 HTML ページの要約を自動的に生成したり，索引付けを自動で行うため，要約の完成度が低くいという欠点がある．

ディレクトリ系検索サービス ディレクトリ系検索サービスは，Yahoo! [14] に代表されるディレクトリー型の検索サービスである．WWW のアドレスを示す URL(Universal Resource Locator) を，芸術，ビジネス，教育...，のように分野別に整理して並べてあるので，分野を決めてから探す時に便利である．データの入力は，基本的に人手で行うため，ロボット系検索サービスに比較してデータ量が 2 〜 3 桁少ないといった欠点を持っている．一方，人手で索引や要約を作成するため，索引と要約の信頼度が高い．

メタ検索サービス メタ検索サービスは，自分自身でデータベースを持たず，ユーザからの検索要求を複数のロボット系検索サービスやディレクトリ系検索サービスに送り，その結果を加工・編集して，ユーザに検索結果として返す検索サービスである．例えば，MetaCrawler [16] では，検索要求を Lycos，Infoseek，WebCrawler，Excite，AltaVista，Yahoo! の 6 つ検索サービスに同時に送り，検

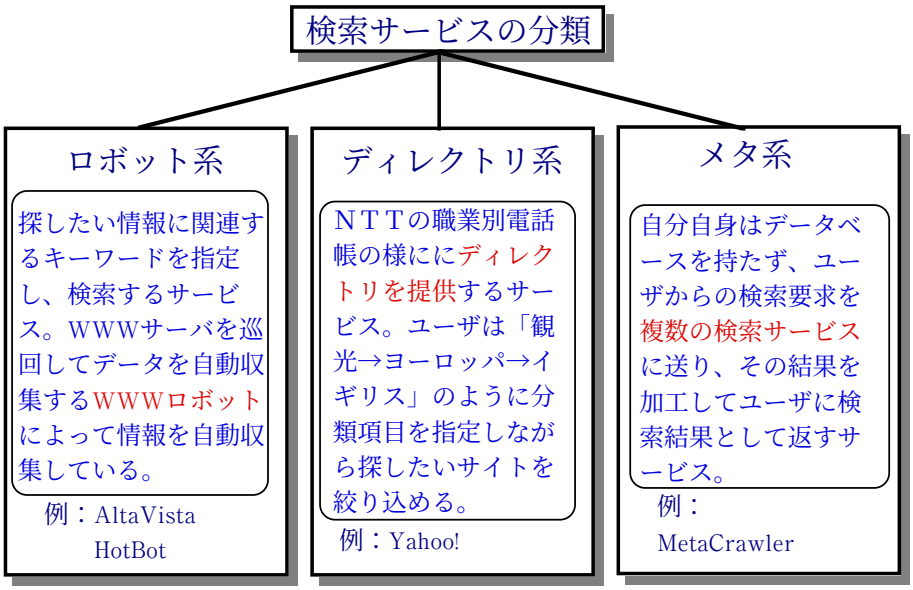


図 2: WWW情報検索サービスの分類

索結果の重複を除去した上で，１つの検索結果のページとしてユーザに返す．このように，メタ検索サービスは，複数の検索サービスの結果をまとめて表示してくれるので便利だが，詳細な検索指定ができない (MetaCrawler では AND,OR,Phrase 検索のみ) と言った欠点もある．

なお，情報検索サービスに関する全般的なサーベイは，[15] をご覧いただきたい．

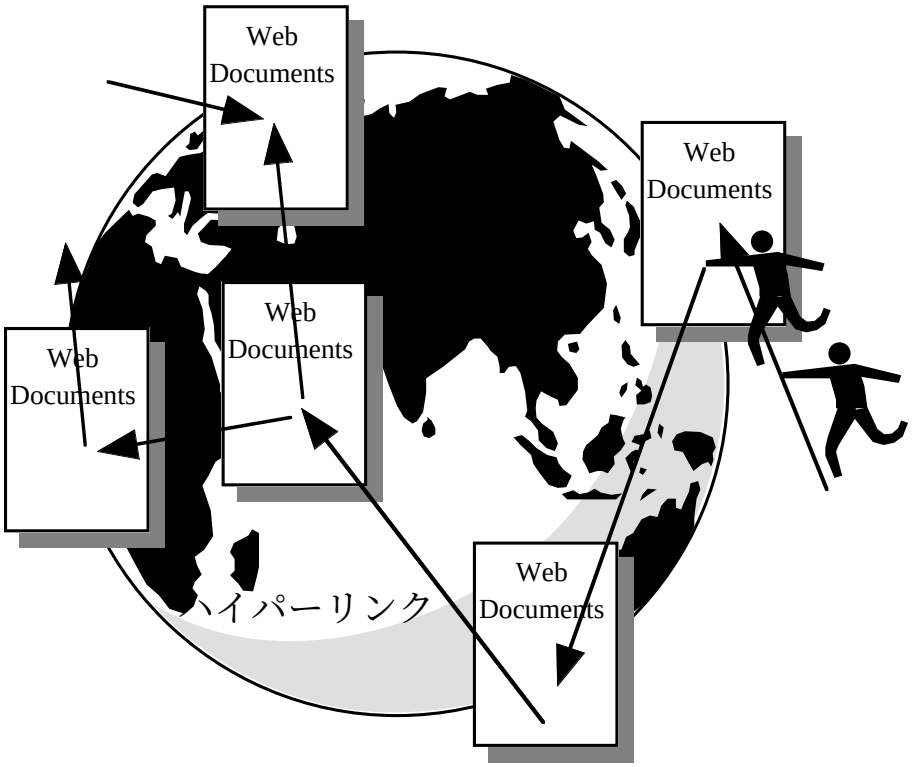
2.3 WWWロボットの問題

ロボット系検索サービスでは，HTTP クライアントの一種である WWW ロボット [3] と呼ばれるソフトウェアを用いて WWW サーバ上のデータを収集し，収集したデータに対する検索を提供している．

WWW ロボットは，次々と WWW サーバにアクセスし，HTML ファイルを自動的に取得する．そして，収集した HTML ファイルに書かれたリンクを辿り，別の WWW ページを取得するという一連の動作を繰り返すことにより，WWW 上のデータを収集している (図 3)．現在の WWW ロボットは，AltaVista や HotBot をはじめとする各々の検索サービスで個別に動かしており，1999 年 2 月現在，世界中で 192 個のロボットが存在する [4]．これらの WWW ロボットは，以下のようなの問題を持っている．

① ネットワークと WWW サーバの負荷問題

現在の WWW ロボットは，検索サービス毎に別々に動作しているため，一つの WWW サーバに対して数十の WWW ロボットが訪れている．このような多数の WWW ロボットによる同一 WWW サーバへのアクセスは，ネットワーク及び WWW サーバに大きな負荷を与えている．



Webロボットはハイパーリンクをたどりながら
Webページを収集する。

図 3: Ｗｅｂロボットの仕組み

② WWW ページ取得時間増大の問題

WWW サーバ数は，図 1に示したように指数関数的に増大しており，全てのデータを一ヶ所に収集し検索システムを提供することは，困難になりつつある．実際に，1996 年の秋以降，商用検索システムの収集 URL 数の増加が鈍化し，現在は約 1 億 2000 万 URL 程度で頭打ちになっている．これは，WWW ロボットを動作させるホスト及び回線容量がボトルネックになっているためである．例えば，早稲田大学で開発した検索サービス「千里眼 [18]」の WWW ロボットは，50 万ページの収集に約 1ヶ月 (実測値) を要している．

③ 収集データ陳腐化の問題

WWW ページは頻繁に更新されるため，最新情報に対する検索を提供するには，WWW ロボットで再度収集して，検索システムの索引を作り直す必要がある．しかし，現在の WWW は，サーバ側からページの更新をクライアントに知らせる機能をもたず，WWW ロボットが，一つ一つのページにアクセスしてみるまでは，更新されているかどうかが分からない．このため，常に最新のデータを検索対象とするには，既に収集されたページについて，更新チェックを頻繁に行わなければならない．例えば，Infoseek [17] では，既に収集したページに対して，1ヶ月後に更新チェックを行い，該当ページが更新されていれば，次は 2 週間後に更新チェックをするといったように，更新チェック間隔

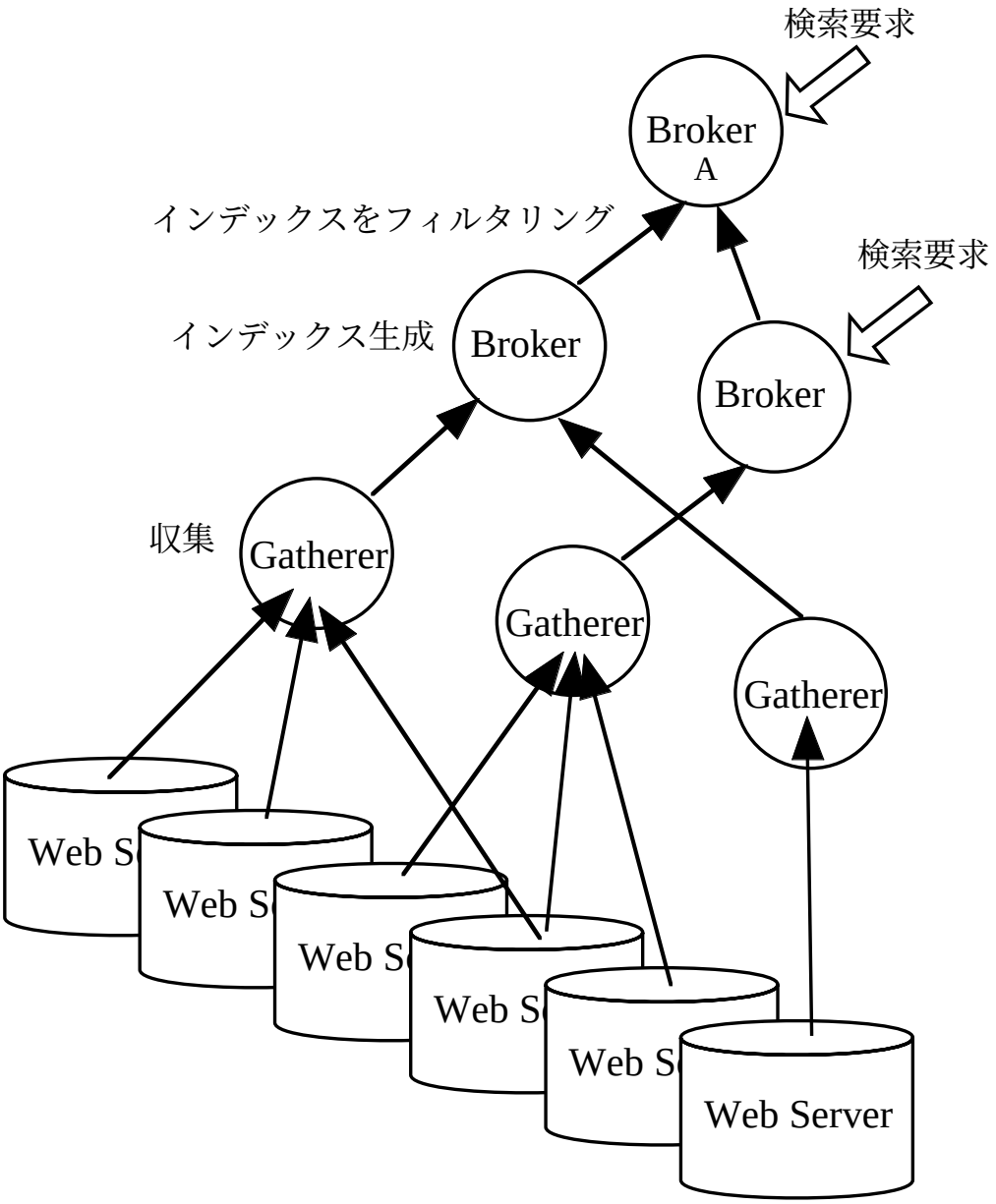
を自動的に変更する方式をとっているが，根本的な解決策ではない．

2.4 分散型WWWロボットの試み

WWW ロボットが持つ問題を解決する一つの方法として，複数の WWW ロボットを分散させ協調して WWW データを収集することによって，WWW サーバへの負荷を軽減すると共に，高速な収集を実現しようとするいくつかの研究がある．以下では，我々の研究開発以外の関連研究について紹介する．

1) Harvest [6] Harvest は，インターネットの将来や現状に関して技術側面から研究を行う団体である IRTF(Internet Research Task Force) [19] の研究グループ (IRTF-RD) が中心となって，ARPA(Advanced Research Projects Agency(現 DARPA) [20] や NSF(National Science Foundation) [21] のサポートを受け 1993 年〜 1996 年に行われた研究である．

Harvest では，Gatherer と呼ぶ WWW ロボットをネットワーク上の複数個所に配置し分散収集をすることができる．また，Broker と呼ぶインタフェースを用いることにより，複数の Gatherer が収集したデータに対する検索ができる (図 4)．しかし，Harvest では，Gatherer を各 WWW サーバ上に (あるいは同一のドメイン内に) 配置し，ネットワーク上の負荷を軽減することを想定していたため，分散収集時の WWW サーバの割り当てに関する研究は行われていない．このため，Gatherer が受け持つ WWW サーバ群は，静的に URL で指定することしかできず，分散を自動的に行うための仕組みを持っていない点が残念である．なお，Harvest で開発された技術は，現在の Netscape のカタログサーバーに利用されている．

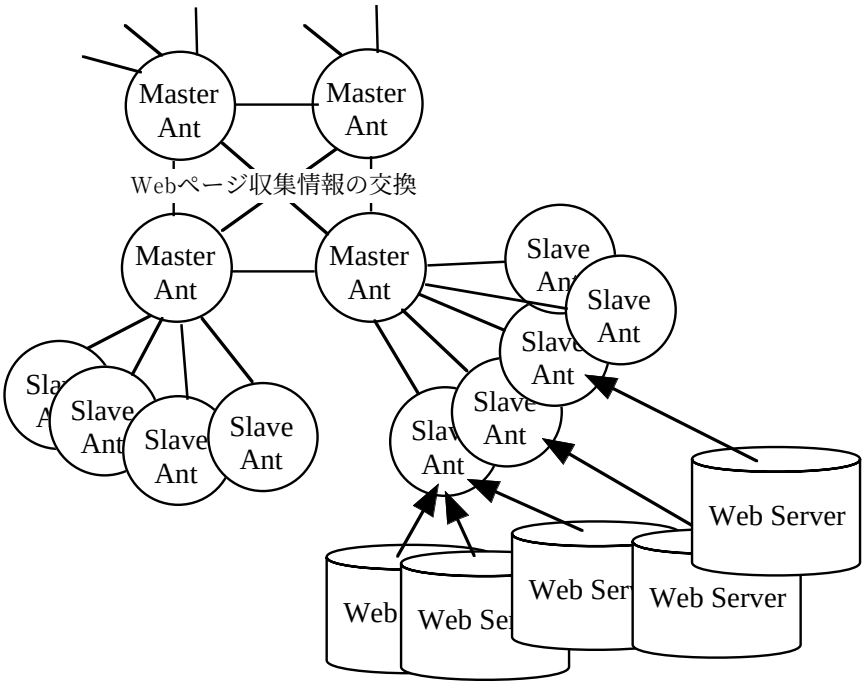


Gathererは複数のWebServerのデータ収集を担当し(重複設定も可能)、収集したデータは、Brokerでインデックス化される。Broker-Aのように他のBrokerの上位にあるBrokerは、下位のBrokerからフィルタリングされたインデックスを受け取ることができる。検索はBrokerに対して行うことができる。

図 4: Harvest:Gatherer と Broker

2) WebAnts [22] WebAnts は，Texas Instruments からの資金を受け，1994 年～1995 年に John R. R. Leavitt 氏 (現在 Lycos) を中心にカーネギーメロン大学で行われたプロジェクトである．

WebAnts は，複数のエージェント (Ant(蟻) と呼んでいる) が協調して WWW サーバ群に対して索引付け (indexing) と検索を行う機能を提供する．索引付けでは，各 slave Ant が Master Ant に対して対象となる Web ドキュメントを収集してよいかどうかを尋ねる．Master Ant は，他の slave Ant で収集していなければ，「収集開始の指示」を出すが既に他の slave Ant で収集していれば「収集不要の指示」を出す．また，master Ant 間ではどの Web ページが収集されているかという情報を共有する (図 5)．このような仕組みによって，Web ページを分散収集する際の重複を避け，効率的な収集が可能となる．WebAnts の具体的な性能についての報告が無いため，性能評価ができないが，1996 年 5 月に検索サービスサイトの Lycos に採用されて現在に至っている．



Slave AntがWebデータを収集する際には、Master Antに「既に収集済みかどうか」を尋ねる。Master Antは他のMater Antとの間でWebデータの収集情報を交換しており、まだ収集していなければ「収集開始の指示」を出す。一方、収集済みであれば「収集不要の指示」を出す。このようにして各Slave Ant間でのWebデータ収集の重複を避けながら分散収集する。

図 5: WebAnts の仕組み

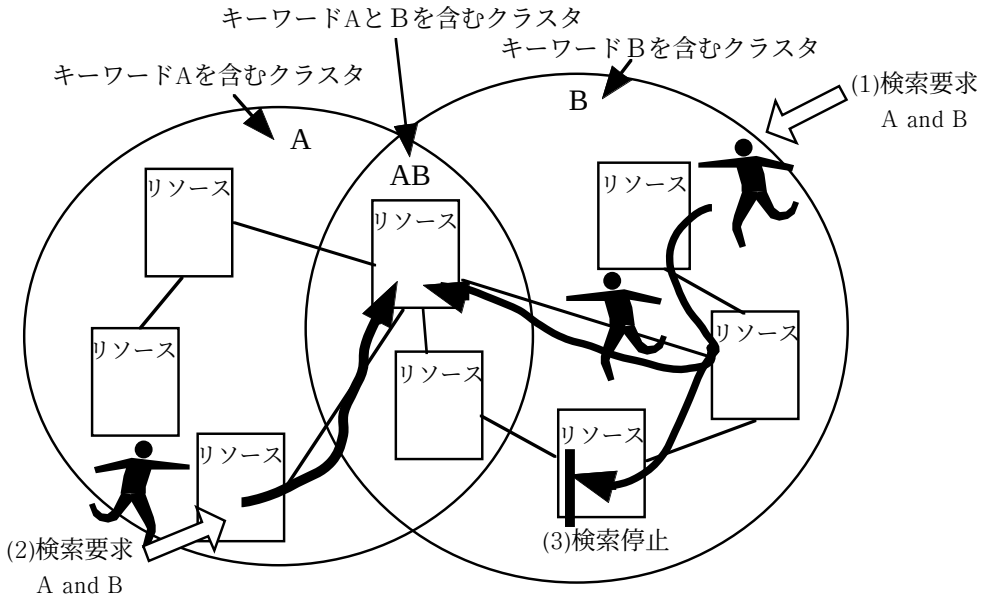
2.5 広域分散協調サーチロボットの未来

ユーザからの膨大な数の検索要求 (例えば，Goo では一日 100 万件以上 [23]) を高速処理するためには，WWW ロボットだけでなく，検索自体も分散させることが求められるようになってきている．

現在の情報検索サービスサイトは，WWW ロボットによって収集したWWW データを一ヶ所に集め，索引付けを行い，検索サービスを提供している．内部的には，複数の計算機 (ワークステーション等) を用い，ユーザからの検索要求を分散させたり，索引 (index) を分散させることによって高速化を図っている．しかし，もともと分散して存在しているWWW データを集中的に扱うのは，ネットワークトラフィックの面から，必ずしも効率がよくない．

一方，もっと根本的に仕組みを変えてしまおうという研究もある．NTT ソフトウェア研究所のIngrid プロジェクト [24] では，WWW ロボットによって，WWW データを収集するのではなく，検索対象 (リソース) が持つキーワードに着目して，図6に示すように，同一のキーワードを持つリソース間に専用のリンクを張り，Ingrid トポロジーと呼ばれる検索のためのトポロジーを構築する．このトポロジーを構築する部分が，従来のWWW ロボットによるWWW データ収集部分に相当する．

検索では，例えば A *and* B の検索の場合，A あるいは B を含む任意の地点から検索がスタートする．スタート地点から，順次Ingrid 専用リンクをたどり A *and* B が満たされるリソースを探す．ひとつでも A *and* B を満たすリソースを探すことができれば，あとは，同一クラスタ (同一のキーワードを持つリソース群) のデータを順次探し，ユーザに返すことで検索が終了する．このように，Ingrid では，検索に前もって検索用のトポロジーを構築しておき，検索の段階で，これまでのWWW ロボットが行っていたような巡回により検索結果を見つけるという方法をとっている．そして，このような手法により，検索自体をIngrid トポロジー内で分散させている．



同一のキーワードを含むリソース間に専用のリンクを張ることにより、Ingridトポロジーを構築する。検索は、検索要求中の任意のキーワードを含む地点、例えば、(1)あるいは(2)からスタートする。リンクをたどることによって、目的のリソースを探す。一定の検索限界(検索の深さなど)に達したものについては、検索を停止させる(上記の(3))

図 6: Ingrid の仕組み

以上，本節では，WWW 検索サービスとインターネット広域分散協調サーチロボットの最近の動向について調査した．インターネット広域分散協調サーチロボットは，指数関数的に増大する WWW 空間の巨大データへの対処法に関する糸口を与えてくれるものであり，今後のさらなる研究が期待される分野でもある．また，次世代の WWW のあるべき姿も，こうした研究から見えてくることが期待される．

3　本研究開発の目標

WWW ロボットが全ての WWW サーバにアクセスして必要な WWW データを収集するには，現状技術では，WWW ロボットを動作させるサーバの性能および回線容量の制限などから，日本国内のみの WWW サーバを対象とした場合でも１ヶ月以上の時間を要する．

このような現状を打破するため，本研究開発では，広域な範囲の WWW データをインターネット上に分散された複数の WWW ロボットで協調して高速収集するための，分散協調サーチロボット（ソフトウェア）の研究開発を行い，日本国内 (ドメイン名が jp で終わる WWW サーバ) の全 WWW データを 24 時間以内に収集することを目標とする．

3.1　ベースとして用いる理論・技術

以下の論文で提案された分散型 WWW ロボットの基礎技術を利用する．

田村健人, 分散 *WWW* ロボットに関する研究, 早稲田大学修士論文 *(1997.3)*

3.2　今年度当初の研究開発計画概略

本研究は平成 10 年度から開始し，平成 11 年度における終了を予定している．平成 10 年度は下記に関する研究開発を実施し，分散協調サーチロボットの実現可能性についてプロトタイプによる評価を行ない，報告書を作成する．

3.2.1　分散協調サーチロボットのプロトコルの検討

多数の WWW ロボットがお互いに協力して分散収集を行なうための，ロボット間協力のための分担・通信などの規則を定める．それぞれのロボットが非同期に収集開始および収集終了をしても，矛盾無く動作できるようにするとともに，新 WWW サーバを発見した時にはこれを通知してこの新たな収集分担を決定するなどのためのプロトコルを設計し，実装する．これらの制御のために，実際の収集を分担する WWW ロボットと，上記のような管理業務を行なう管理サーバを設けて，これらの間で作業分担を行なう方式を採用する．

3.2.2　分散エリア決定方法の検討

各 WWW ロボットの分担エリアを決定する方式について検討し，設計に反映させる．収集範囲の決定には，WWW ロボットと WWW サーバ間のデータ伝送時間と実際に伝送すべきデータ量を使用する．データ伝送時間としては，ping 時間でもって代替することにして，これらの必要データ収集の効率化を図ることとする．各 WWW ロボットの分担範囲は，それぞれのロボットが伝送すべき WWW ページ総データの伝送時間がロボット間でほぼ均等になるようなアルゴリズムを採用する．

3.2.3　分散協調サーチロボットプロトタイプシステムの設計と先行評価

プロトタイプシステムの設計を検討すると共に，試作を進めて，その結果を国内複数サイトの協力を得て，先行評価する．具体的には，国内に50個所の評価サイトを選択して，プロトタイプシステムをこれらの評価サイトに配備して，分散協調収集の実験を行なう．これら50個所の分散協調サーチロボットを使用して，分散協調サーチの効果を評価するためのデータを収集する．評価用データとしては，収集速度の向上効果，実時間性などである．また，この評価システムでWWWホームページデータを収集した結果を利用者サイト（すなわち，検索サービス構築者）に配布するような，WWWホームページデータ配布サービスの実現可能性についても評価を行なう．

4　平成10年度研究開発実施報告

早稲田大学 理工学部 村岡研究室，日本アイビーエム株式会社東京基礎研究所 ネットワーク＆ソリューション，京都大学大学院情報学研究科システム科学専攻 河野研究室，電子総合研究所 情報アーキテクチャ部の各研究者が主として，研究統括，ソフトウェア開発，アルゴリズム開発に携わるとともに，慶應義塾大学 環境情報学部，北陸先端科学技術大学院大学，大阪府立大学工学部，シャープ（株）技術本部 ソフトウェア研究所の協力を得て，研究を実施した．なお，ここに挙げた以外にも，付録１に示す多くの大学，企業，ネットワークプロバイダ，ネットワーク協議会の協力を得ている．

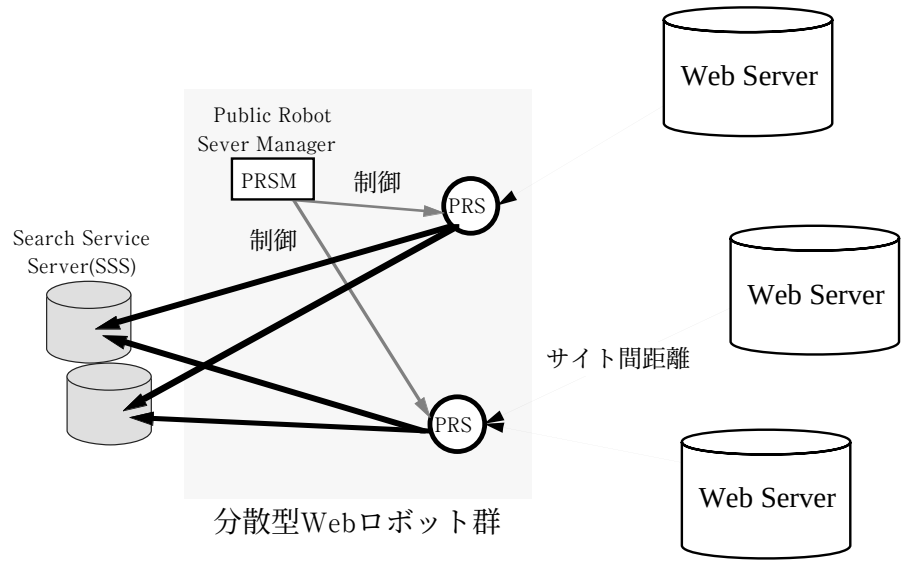
以下に今年度の研究開発計画の項目毎に実施内容を詳細に示すと共に，当初計画との比較を示す．

4.1 分散協調サーチロボットのプロトコルの検討

当初の計画通り，多数のWWWロボットがお互いに協力して分散収集を行なうための，分担・通信プロトコルの設計を行った．また，プロトコルを実装するロボットを試作することにより，プロトコルとしての実用性を確認した．具体的には，分散されたWWWロボットを管理するためのサーバ PRSM (Public Robot Server Manager)，及び，PRS (Public Robot Server) の間のプロトコルを設計しまとめた．以下に詳細を示す．

4.1.1 モデル説明

分散型WWWロボットは，図7に示すように，全体を管理するPublic Robot Server Manager(PRSM) と個々のWWWロボットであるPublic Robot Server(PRS) から構成される．PRSMは，PRSに対して担当WWWサーバの分配や，各WWWサーバとPRS間との距離計測を指示する．一方，PRSで新規に発見されたWWWサーバや距離計測の結果は，PRSMに送られる．このようにして，PRSはPRSMからの指示に基づき各々互いに重複しないWWWサーバを担当しWWWデータを収集する．



PRSは複数のWebServerのデータ収集を担当し(重複設定も可能)、収集したデータを圧縮し一つのファイルにした上で、検索サービスを提供するSearch Service Serverへ送る。PRSの制御は、現在のインプリメントでは、PRSMが集中管理を行う。

図 7: 分散型WWWロボットの仕組み

収集されたデータは，最終的に図中のSearch Service Server(SSS) に再配布することにより，検索サービスのための索引作成を行う．現在のHTTP/1.1準拠のサーバは，1回の接続で複数のデータを転送するKeep-Alive機能を持っているが，複数のデータをまとめて転送する機能は持っていない．このため，各PRSで収集したデータをSSSに再配布する際のオーバヘッドを小さくするため，複数のデータを1回の

接続でまとめて圧縮して送る機能，及び，SSS からの要求に応じてデータ中の必要な部分のみ (例:URL や更新日時の指定による指定) を送る機能を PRS に持たせる.

4.1.2 PRS・PRSM の動作とプロトコル設計

分散型 WWW ロボットである PRS (Public Robot Server) と，PRS 群を制御する PRSM (Public Robot Server Manager) との間のプロトコルを以下の５つに分類し設計した．５つのプロトコルは，以下の通りである.

(P1) 担当するサーバのリスト配布プロトコル (PRSM → PRS)

(P2) 発見したサーバのリスト報告プロトコル (PRS → PRSM)

(P3) 収集ログファイル転送プロトコル (PRS → PRSM)

(P4) リスタート要求プロトコル (PRSM → PRS)

(P5) 参加・脱退プロトコル (PRS → PRSM)

本システムにおける PRS と PRSM の動きは，以下の通りである.

① PRSM は各 PRS に各々の担当サーバリスト (P1) を送信

② 各 PRS は担当サーバリストに従い WWW ページを収集

③ 各 PRS は収集 WWW ページを解析し未知の WWW サーバを PRSM に通知 (P2)

④ 各 PRS は WWW ページ収集情報 (サイズ, 転送時間) を PRSM に通知 (P3)

⑤ PRSM は各 PRS から送られた情報を元に，次の担当サーバリストを作成

PRS がファイアウォール内に設置される場合も考慮し，PRS・PRSM 間の通信には HTTP を採用した．さらに，PRS からの PRSM に対するポーリング方式を採用することにより，PRS と PRSM との間の非同期性を確保した．P4. は，評価実験を効率的に行うためにに付加されたプロトコルであり，PRSM 側から強制的に PRS の担当サーバを変更することができる.

4.1.3 PRS・PRSM 間プロトコルの詳細

以下の説明で，*PRSM* は PRSM のホスト名，*PRSNAME* はアクセスする PRS のホスト名である．

(P1) 担当するサーバのリスト配布プロトコル (PRSM → PRS)

`http://`*PRSM*`/servlets/list?host=`*PRSNAME*

PRS は PRSM の特定の URL にアクセスし，返答として担当サーバのリストを取得する．担当サーバリストは，URL もしくは URL とその URL のリンク元 URL を 1 行に書いたもの (例: www.etl.go.jp:80) で，それを gzip で圧縮して転送する．以下にデータ形式を示す．

```
     ServerList  ::=  gzip(ServerListItem*)
 ServerListItem  ::=  URL (HT RefererURL)? LF
            URL  ::=  <URL; RFC2396>
     RefererURL  ::=  <URL; RFC2396>
             HT  ::=  <US-ASCII HT, horizontal-tab (9)>
             LF  ::=  <US-ASCII LF, linefeed (10)>
```

(P2) 発見したサーバのリスト報告プロトコル (PRS → PRSM)

`http://`*PRSM*`/servlets/record`

PRS は PRSM の特定の URL にアクセスし，HTTP の POST メソッドにより発見サーバリストを送信する．形式は担当サーバリストとまったく同一である．gzip で圧縮する点も同じである．

```
 NewServerList  ::=  gzip(Server*)
```

(P3) 収集ログファイル転送プロトコル (PRS → PRSM)

`http://`*PRSM*`/servlets/submitlog`

PRS は PRSM の特定の URL にアクセスし，HTTP の POST メソッドにより収集ログを送信する．形式は第 1 行目 (最初の LF まで) がファイル名で，それ以降はログを gzip 圧縮したものである．

```
      Log  ::=  FileName LF gzip( (LogLine | Comment)* )
  Comment  ::=  '#'[LF]*LF
 FileName  ::=  PRSNAME '.' 日付
  LogLine  ::=  '"' URL '",' HTTP 要求サイズ ',' HTTP 応答サイズ ',
                ' 所要時間 ',"' HTTP 応答ヘッダの中身 '"' LF
```

(P4) リスタート要求プロトコル (PRSM → PRS)

実験の便を考慮し，前述のプロトコルに，「リスタート要求」を加えた．PRS は 1 時間毎に `http://`*PRSM*`/servlets/isrestart?host=`*PRSNAME* にアクセスし，その返答の 1 行目が "`RESTART`" の場合，現在の担当サーバリストの破棄・ディスク上にある収集したページの削除を行ない，あらたに担当サーバリストを取得する．

(P5) 参加・脱退プロトコル (PRS → PRSM)

`http://`*PRSM*`/servlets/subscribe?host=`*PRSNAME*

`http://`*PRSM*`/servlets/unsubscribe?host=`*PRSNAME*

PRS が参加するとき・脱退するときは上記のような URL にアクセスする．

4.1.4 PRS・PRSM のプログラム仕様

PRS・PRSM の主要部分を Java で実装した．PRS は完全な Java アプリケーションである．PRSM はウェブサーバ Apache に JServ モジュールを組み込み，処理は Java servlets で行う．

4.1.5 PRS の動作

PRS は，まず起動時に PRSM にアクセスし，担当リストを PRSM から取得する．担当のサーバのページをすべて取得し終ると，再び担当リストを取得する．

ページ収集は時間を制限している．平日は午前 2 時〜午前 8 時，土日は午前 2 時〜午後 10 時のみページ収集を行う．

同時に 100 個のサーバに対してアクセスを行なう．1 つのサーバに対しては 20 秒おきにアクセスを行なう．ただし，ウェブサーバが HTTP/1.1 の Keep Alive 機能に対応している場合は可能な限り連続でアクセスを行なう．

一日に一度，午前 9 時ごろ[1]，収集ログを PRSM に転送する．

4.1.6 プロトタイプインプリメント上の現状報告

上記のプロトコル仕様に基づき，PRS と PRSM のプロトタイプを試作した．プロトタイプでは，いくつかの動作上の制限があるので，以下に詳細を示す．

- 本来は，PRS から PRSM へ収集ログを転送した後，PRSM が集計を行ない担当サーバリストの更新を行なう設計であるが，計算部分との統合はまだなされていない．これは，計算部分 (分散アルゴリズム) を次節で示すように検討

[1] PRS は起動後 20 分毎に現時間をチェックしており，午前 9 時を過ぎた時点 (従って午前 9 時〜午前 9 時 20 分の間) で本処理を実行する．

中であるためである．従って，現時点では，次節のアルゴリズムを適用して得られた担当サーバリストを手動でPRSMに設定している．

- 上記と同様の理由から現在はPRSMがPRSから新規発見サーバリストを受け取っても，何ら処理をしていない．
- プロトコル上は，担当サーバリスト・発見サーバリストはサーバのトップページだけでなく任意のページを入れることができる．また，referer情報(そのページを発見することになったリンク元のURL)を含めることが可能である．しかし，情報交換によるトラフィック増加を考え，実装では「サーバのトップページのみ・refererなし」でやりとりしている．これにより，現在のPRSは，あるサーバのトップページからそのサーバ内のリンクだけで辿れないページを収集することができない．トップページだけでなく，すべてのページの情報を送受信した場合にどの程度のトラフィックが発生するかを検証すべきである．
- PRSの動作環境は，今回はすべてSolaris 2.6 Intel Editionであるが，PRSのプログラムをJava 1.1のJava Virtual Machine (JVM)で動作させると数十分後にJVMがエラーを出して停止するという現象が起きた．この問題はJava 2 β版のJVMでは発生しない．PRSのプログラムはJava 1.1の機能のみで書かれているが，この問題を回避するため実験ではJava 2 β版を使用した．

4.2 分散エリア決定方法の検討 [25]

当初の計画通り，各WWW ロボットの分担エリアを決定する方式について検討を行った．各WWW ロボットの負荷を均等化するための「負荷均等化アルゴリズム」を提案し，分散を求めるためのプロトタイプシステムを作成した．そして，実データを用いて負荷均等化が実際に有効であることを確認した．以下に詳細を示す．

なお，当初の予定では，負荷均等化を行うために，ping コマンドによる時間を利用する予定であった．しかし，検討過程で，ping コマンドよりも，実際に WWW サーバにアクセスし一定のファイル (50 ファイル) を http により転送して転送速度を求める手法の方が正確であることが判明した．このため，転送時間の算出に，http プロトコルによる一定のファイル転送を行い，利用することとした．

4.2.1 Web 空間のモデル化

Web 空間はHTML で記述されたドキュメントがリンクという形で相互に，あるいは一方的に結ばれた仮想的な空間である．実際にはHTML ドキュメントを保持するWeb サーバ（ホスト）が回線によって物理的に結びつけられた空間であり，インターネットの部分集合をなしている．これは図8に示した次の構成要素からなる．

- HTML で記述された各Web ページ
- Web サーバが動作している計算機（ホスト）
- インターネット

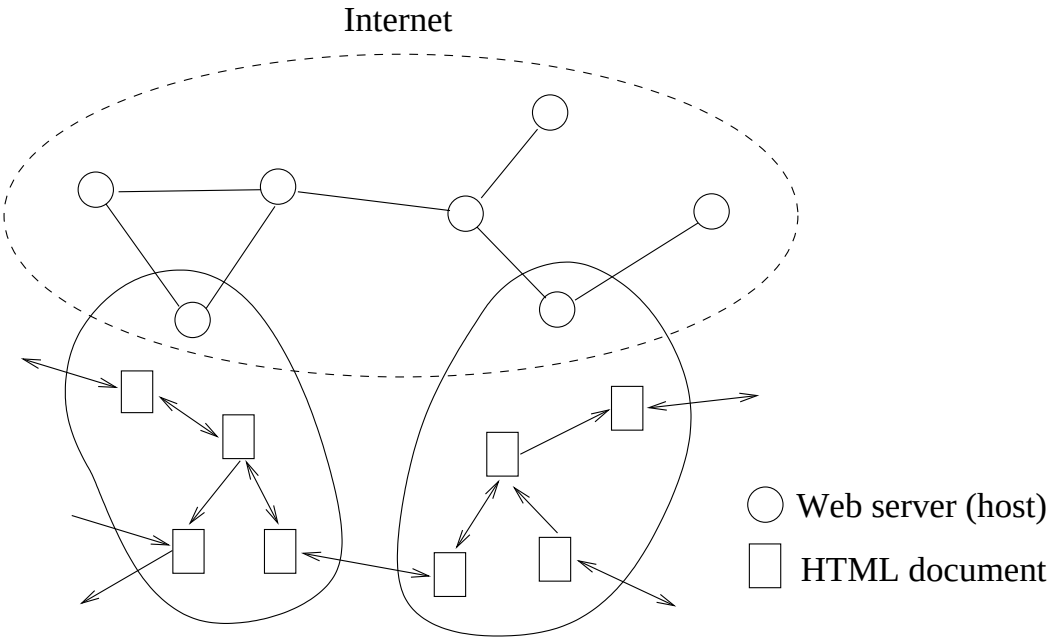


図 8: Web 空間

ここで収集の主な対象となるのはHTML で記述されたWeb ページであるが，膨大なページをページ単位で収集するのではなく，ネットワークの物理的接続を考え

てより大きな単位で収集するものとする．なお，小規模な分散協調型ロボットの場合は，ドメイン単位での収集も十分に好ましい性能を確保できることが分かっているが，本研究ではスケーラビリティを考えたシステムとするためより小さな単位としてホストを対象とする．したがって，これによりWeb空間はホストを接点とした連結有向グラフによって表現できることになる．

4.2.2 ロボットのモデル化

ロボット [3] とはHTTPクライアントの一種であり，Webページを自動的に回収するソフトウェアのことである．WWWロボットとWebブラウザとの違いは能動的，自律的に動作するかどうかである．検索エンジン用のロボットの場合は，次々とWebサーバにアクセスしては，HTMLドキュメントを取得し，またそこに書かれているリンクを辿って，別のWebページを取得するという動作を繰り返す．

Web空間上にロボットプログラムが置かれたサイト（以下簡単のため「ロボット」と表記する）を複数設置し，分担して収集するホストを担当する．ただしロボット間で担当ホストに重複はないものとする．

各ロボットには，収集を行うことにより

- 自分が収集を担当しているホストのリスト
- 収集したWebページのデータ
- 各ホストのドキュメント量 (KByte)
- 各ホストの収集に要した時間 (msec)

などの情報が保持されている．

なお，ロボット間で互いに情報交換を行いながら負荷を均一化するプロセスを実行することになるが，各々のロボットが他のすべてのロボットと情報交換を行うと，ロボットの数が増えるにつれてメッセージ量が増大して効率の低下が予想される．そこで，ロボット間の情報交換にもある程度の制約を与えることを考える．さらに，ネットワークに与える負荷を考慮するならば，一般にネットワークルーチングプロトコル [26] と同様に，距離が近いロボット同士で局所的な情報交換を行うほうが望ましいと考えられる．

そこで，上記の条件をみたすための方法として各ロボット間のネットワーク距離を考慮したコストを用いて最小木を与え，最小木に沿う隣接ロボット間でWebデータ収集情報の交換を行うことにする．すなわち，ロボットを接点，ロボット間の距離を枝の重みとし，プリムのアルゴリズム [27] によって最小木を作り，図9のように最小木に沿って情報交換を行うものとする．

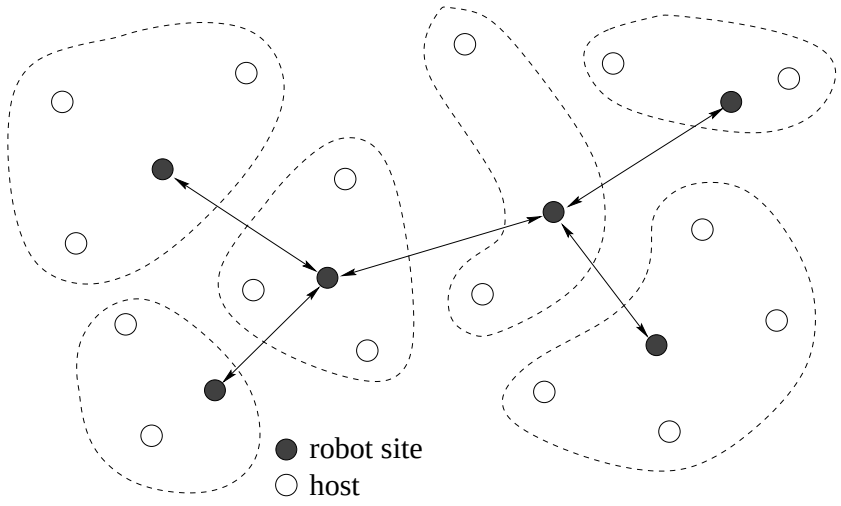


図 9: 最小木に沿った隣接ロボット間の情報交換.

4.2.3 パラメータの定義

Web 空間にロボットがmヶ所あるとし，その集合を

$$R = (r_1, r_2, ..., r_m) \tag{1}$$

とし，ロボットは最小木をなしているとする．また収集対象となるホストがnヶ所あるとし，その集合を

$$H = (h_1, h_2, ..., h_n) \tag{2}$$

とする．ロボットはこれらのホストを分担して収集する.

4.2.4 コストの定義

ロボットのコストは，自分が担当しているホストに属する Web ページを収集するのに要した時間で定義する．すなわちロボットr_iの担当ホスト$H_i = (h_{i1}, h_{i2}, ..., h_{in_i})$に属する Web ページを収集するのに要した時間をそれぞれ$t_{ij}(j = 1, 2, ..., n_i)$（単位は msec）とすると，ロボット$r_i$のコスト$Cost(r_i)$は,

$$Cost(r_i) = \sum_{j=1}^{n_i} t_{ij} \quad (i = 1, ..., m) \tag{3}$$

となる．先に述べたように担当ホストは重複がないように割り付けるので,

$$H = \bigcup_{i=1}^{m} H_i \tag{4}$$

$$\forall(i, j), i \neq j, H_i \bigcap H_j = \emptyset \tag{5}$$

が成り立つ.

もっとも，ロボットのネットワーク的な位置関係，担当ホスト数の違いなど様々な要因により，各ロボットのコストは異なってくる．そこで最短時間でページを収集することを目標にしながら，収集コストを均等化することを目指す.

4.2.5 距離の定義

本節で述べる距離とは，物理的な距離ではなくネットワーク的な距離である．もっとも，ネットワーク的距離も様々な定義が可能であるが，本研究では，ある一定量のデータを送信するのに要した時間を距離として用いることにする．

なお，ロボットには収集したホストのドキュメント量と，そのホストの収集に要した時間が保持されている．そこでこれらを用いて，

$$\frac{\text{収集に要した時間(msec)}}{\text{ドキュメント量(KByte)}} \tag{6}$$

を計算する．これは1KByteを収集するのに要した時間(msec/KByte)を表しており，先ほど定義した距離と同義である．したがって式(6)をロボットとホストとの距離とする．またこれによりコストは，

$$Cost(r_i) = \sum_{j=1}^{n_i} w_{ij} \cdot d(r_i, h_{ij}) \tag{7}$$

と定義することもできる．ただし，w_{ij}はホストh_{ij}に属するWebページの総ドキュメント量，$d(r_i, r_{ij})$はロボットr_iからホストh_{ij}までの距離である．

4.2.6 アルゴリズム

ロボットによる最小木の構築 まずプリムのアルゴリズムを用いてロボット間で最小木を作る．連結無向グラフ$G = (V, E)$において，最小木を求める方法は以下のとおり．

プリムのアルゴリズム

1. 適当な節点$u \in V$を選び，node-set $:= \{u\}$，edge-set $:= \{(u, v) | (u, v) \in E\}$，tree $:= \emptyset$とする．
2. node-set $= V$となるまで以下の3,4を繰り返す．
3. edge-setの中で最小の長さをもつ枝(u, v)を選ぶ．（ただし，$u \in$ node-set, $v \notin$ node-set）
4. tree $:=$ tree $\cup \{u\}$
   node-set $:=$ node-set $\cup \{v\}$
   edge-set $:=$ edge-set$\cup\{(v, w) | w \notin$ node-set, $(v, w) \in E\} - \{(y, v) | y \in$ node-set, $(y, v) \in E\}$
5. treeを出力する．treeはGの最小木の枝集合を与える．

ロボット間の距離を求めるには式(6)を用いればよい．つまり収集するホストのリストにロボット，あるいはそれに近いと思われるホストを加えて収集を行えば距離を求めることができる．

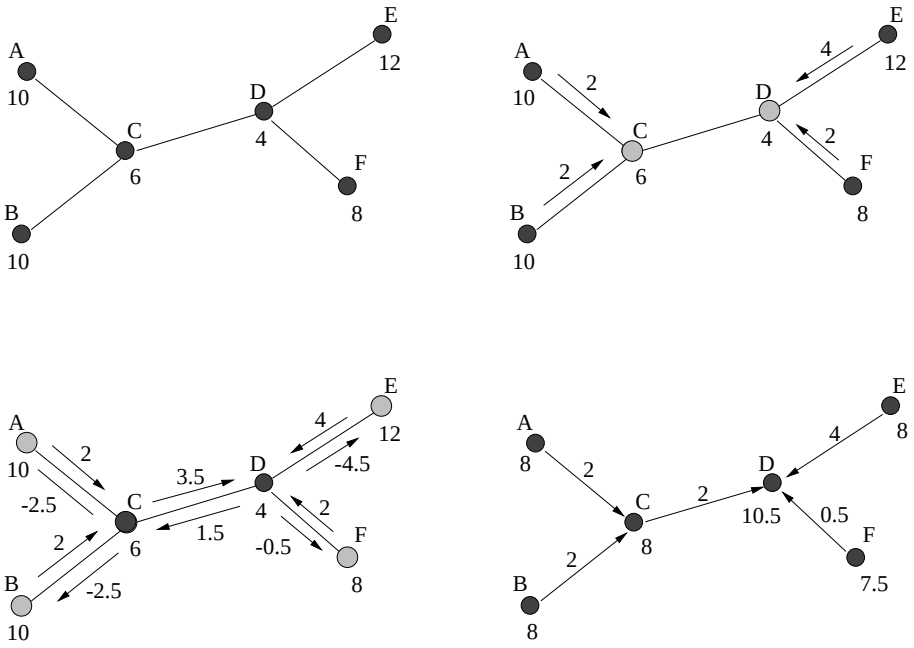


図 10: 均等化の例.

4.2.7 隣接ロボット間のコスト均等化

各ロボットは自分が担当しているホストの収集を終えた後コストを計算し，最小木に沿って隣接ロボット間でコストの均等化を行う．具体的には隣接ロボット間で担当ホストおよびそのホストのWebデータの受け渡しをすることで均等化を行う．

なお，以下では，図10の例を用いて，提案するアルゴリズムを説明することにする．また，各々のロボットに付された数字は，そのロボットのもつWebページに対するコストを表すものである．なお，簡単のために，次のような表記を用いることにする．

$Move(A, B)$ ： ロボットAからロボットBへ移動させるコスト量.
この値が正ならコストを渡し，負ならコストを受け取る
ことを意味する.

まず，ロボットAについて考えると，隣接ロボットはCであり，コストはそれぞれ10，6となっているので，互いのコストを均等化することを考えると$(10+6)/2=8$となる．つまり$Move(A, C)=8-6=2$である．ロボットB，E，Fについても同様にして移動コスト量を求めると図10右上のようになる．

次にロボットCについて考える．隣接ロボットはA，B，およびDであり，コストはそれぞれ6，10，10，4であるからこれを均等化すると$(6+10+10+4)/4=7.5$となる．したがって$Move(C, A)=Move(C, B)=-2.5$，$Move(C, D)=3.5$と求められる．ロボットDについても同様にして移動コスト量を求めると図10左下のように

なる．

ここでA，C間を見てみると，Aは$Move(A,C)=2$より，Cに対してコスト2だけ渡したい．一方Cは$Move(C,A)=-2.5$より，Aに対してコスト2.5まで受け取ることができる．したがってA，C間の正味のコストの移動量はAからCへ2となる．

次にD，F間を見てみると，FはDにコスト2だけ渡したいのだが，Dはロボットから0.5しか受け取ることができないので，結局正味のコスト移動量はFからDへ0.5となる．

またC，D間では$Move(C,D)=3.5, Move(D,C)=1.5$となっていてお互いがコストを渡そうとしている．このような場合には相殺しあって$Move(C,D)=2$とする．お互いが受け取ろうとする場合も同様に考える．

以上のような操作をすべてのロボットが行うと図10右下のようになる．ロボットに付された数字は均等化後のコスト，枝の重みは正味のコスト移動量である．ただしここでのコストは移動元のコストを基準にして計算したものである．つまり今の例ではAからCへコストが2だけ移動しているが，この2はAにとってのコストであり，実際のコストがどうなるのかは新たに収集を行うまでわからない．

コスト均等化の留意点　上記の方法は，コストの均等化に注目していてどのホストを渡すべきなのかを考えていない．式(7)よりコストはロボットとホストとの距離に依存するので，

1. 隣接ロボットにホストを渡した結果，図11のようにホストとロボットとの距離が渡す前よりも長くなり，かえってコストが増加してしまう．
2. また，そのようなホストは隣接ロボットに渡さないようにしたとしても，図12のように複数ステップ先のロボットでは距離が短くなる場合も考えられる．当然そのロボットに担当させたほうがコストも小さくなる．

といった問題点を挙げることができる．したがってホストを受け渡しする際には上記の方法に加え，ロボットとホストの距離を考慮しなければならない．さらに2のような場合も考えると，隣接ロボットだけでなく複数ステップ先のロボットとホストとの距離も知る必要がある．

そこでホストを渡す前に，複数ステップ先までのロボットに対し，そのホストまでの距離を測定するよう依頼し，その距離が最も小さいロボットに通じる隣接ロボットにホストを渡すことにした．ただし自分がそのホストに一番近い場合は渡さないことにする．距離を求める方法は，本稿では，あらかじめホストに属するWebページの一部（例えばトップページ）を収集して式(6)により距離を計算するなどが考えられる．

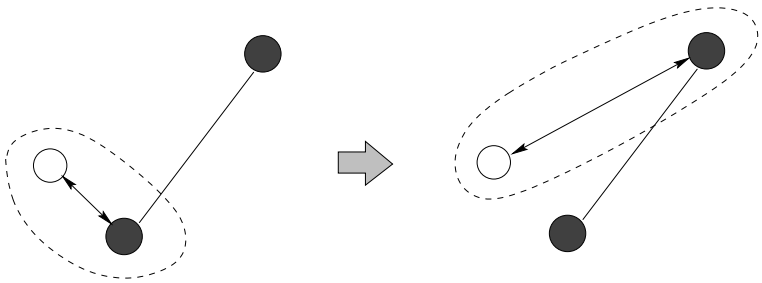


図 11: 隣接ロボット距離が長くなる場合.

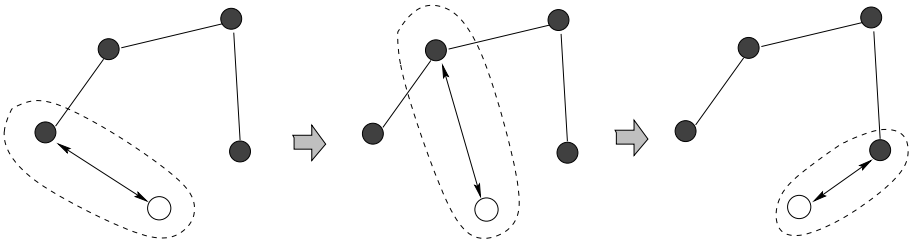


図 12: 複数ステップ先のロボットで距離が短くなる場合

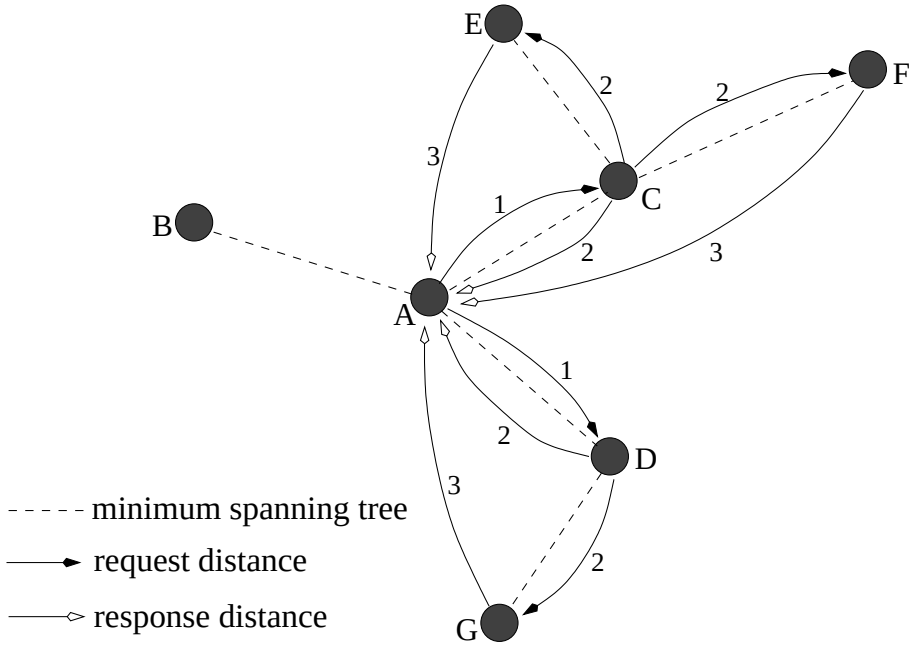


図 13: 距離を考慮した転送の例（ステップ数が 2 の場合）

例えば図13のような場合を考える．今ロボットAは，隣接ロボット間のコスト均等化によりCとDにホストを渡そうとしている．またこの例では距離を測定するステップ数を2とする．

1. AはCとDにホストまでの距離を測定するよう依頼する．
2. BとCは測定した距離をAに返す．また，ステップ数が2より，Cは自分の隣接ロボットであるEとFに，DはGに距離測定の依頼を転送する．
3. 依頼を受け取ったロボットは距離をAに返す．

以上の手順により，距離測定の依頼は最小木に沿って複数ステップ先まで到達し，ロボットAは受け取った結果をもとにホストの転送先を決定する．もしC,E, あるいはFの距離が最小ならばCに転送し，DあるいはGが最小ならDに転送する．ただし自分が最小ならば転送は行わない．

例えばFが最小の場合，Cに転送することによって一時的にコストが増加する可能性があるが，以下に述べるように均等化は繰り返し行うので，次の段階の均等化で転送したホストはCからFへ移動する．

負荷均等化アルゴリズム

初期段階ではホストをどのロボットに割り当てればよいのかわからないので．ランダムに割り当てて収集を行う．収集終了後，先のアルゴリズムで隣接ロボット同士でコスト均等化を行い，新たな分担ホストのもとで再び収集を行う．そして収集終了後，コスト均等化を行い…という動作を繰り返す．

以上をまとめると，負荷均等化アルゴリズムは以下のようになる．

1. ロボット間で最小木を作る．
2. ホストを各ロボットにランダムに割り当てる．
3. 各ロボットは割り当てられたホストを収集する．
4. 各ロボットは隣接ロボット間でコスト均等化を行う．ホストを渡す際は，以下の手順で転送先を決定する．
   (a) n ステップ先のロボットに対し，そのホストまでの距離を測定するよう依頼する．
   (b) その距離が最小となるロボットへ通じる隣接ロボットにホストを渡す．ただし自分が最小の場合は渡さないものとする．
5. 3 に戻る．

負荷均等化の流れをまとまめると図14のようになる．繰り返し均等化を行うことにより，ネットワークトポロジの変化にも柔軟に対応できると考えられる．

なお，距離の測定を依頼するステップ数は，ロボットの数や最小木の形状，収集ホストの数といったシステムの規模によって様々な状況が考えられるが，本稿では，「最小木における最長の枝の長さの半分」とした．

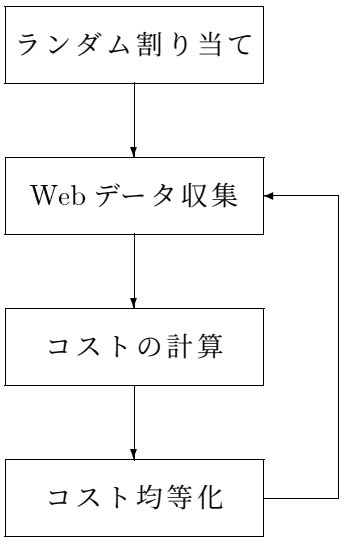


図 14: 均等化の流れ

4.3 分散協調サーチロボットプロトタイプシステムの設計と先行評価

4.1 ～ 4.2 の機能を持つ分散協調サーチロボットプロトタイプシステムを設計し，7 個所の PRS に分散させ，先行評価を行った．また，収集したデータを配布する WWW データ配布サービスの実現可能性についても調査するため，7 個所の PRS で収集した WWW データを一カ所に収集するためのプロトタイプシステムを設計した．

先行評価では，103 の WWW サーバを対象に，7PRS で同一データを収集した場合，7PRS にランダムに分散した場合，7PRS に提案する負荷均等化アルゴリズムで分散した場合の実際の収集時間を測定した．その結果，一カ所で集中して収集する場合に比較し，ランダムな分散をした場合，2.6 ～ 10.6 倍の高速化が可能であり，負荷均一化を行った場合，5.5 ～ 22 倍の高速化が可能であることがわかり，分散協調サーチロボットの有効性を確認することができた．なお，ランダムな分散では，分散しない場合に比較して逆に収集時間が増大する場合が存在することもわかった．

なお，当初の予定では，プロトタイプシステムを利用し，国内 50 個所程度の評価サイトに対してプロトタイプシステムの配備し，先行評価する予定であったが，下記理由により 1999 年 2 月現在の評価は 7 サイトによる評価にとどまる．

JAVA 処理系のバグによるプロトタイプ開発の遅延　プラットフォーム非依存のソフトウェアを開発し公共性を高めるため JAVA 処理系の最新版である JDK (Java Development Kit) 1.1.X を利用した．しかし，Java1.1 の Java Virtual machine (JVM) 上でプロトタイプシステムを動作させると実行開始の数十分後に JVM のエラーにより実行が停止することが判明した．これは，JDK のスレッド処理のバグであり，既に米国サン・マイクロシステムズに報告済みである．これを受けて，サンは，1998 年 12 月 8 日に Java2 β 版を公開した．この β 版を利用することにより，上記 JDK のバグが解消されたが，プロトタイプシステム自身のデバッグ作業開始が大幅に遅れた．

広域分散環境下でのデバッグ作業の困難性　Java 自身が持つバグのため，開発ソフトウェアのデバッグ作業が開始されたのは 1998 年 12 月 8 日である．その後，まず，プロトタイプの品質を向上させるため，早稲田内に設置された 4 台のマシンで半月間デバッグを行った．その後，早稲田, 電総研, シャープ, 京大, 府立大, 北陸先端大, 慶應の 7 サイトにプロトタイプを設置し，試験運用を開始した．しかし，インターネットの不安定さに起因して発生する予期不可能な状況 (PRSM と PRS が通信中にネットワークダウン等) によるバグ発生のため，デバッグ作業が長期化した．安定的に動作する状態になったのは，デバッグ開始して 2ヶ月後の 1999 年 2 月 12 日である．以上のように，広域に分散された環境で，このような分散協調処理型ソフトウェアのデバッグが極めて困難を要することが認識された．

システムの自動バージョンアップシステム整備を優先　全国32サイト(19大学,5プロバイダ,3企業, 3インターネット協議会,2国研)の協力を得て(http://www.etl.go.jp/~yamana/DWR/dwr-members-open.html)，32サイトにJava 2をインストールし，プロトタイプのインストール及び実行を開始できる状態にある．しかし，インストールやプロトタイプのバージョンアップに伴う作業を32個所で個別に人手で行うのは，極めて困難を要する．したがって，これらの32サイトに対してインターネット経由でのプロトタイプの自動配送・自動インストールするための技術開発(セキュリティ対策を含む)が，プロトタイプ配布の前に必要不可欠であるとの結論に達した．

4.3.1　実験結果及び考察

初期設定

まず表1に本実験に参加した組織と，ロボットプログラムを設置したサイト（ロボットサイト）を示す．また，これらのロボットサイトで構成した最小木を図15に示す．ただしここでの節点の位置は便宜的なものであって，地理的条件とは関係ない．なお，最小木の最長の長さが5であるから，距離の測定を依頼するステップ数を3とした．

表 1: 実験に参加したロボットサイト（順不同）

組織名	ロボットサイト
京都大学	XXX.YYY.kyoto-u.ac.jp
早稲田大学	XXX.YYY.info.waseda.ac.jp
電子技術総合研究所	XXX.etl.go.jp
シャープ株式会社	210.224.XXX.YYY
大阪府立大学	157.16.XXX.YYY
慶應義塾大学	XXX.YYY.wide.ad.jp
北陸先端科学技術大学院大学	XXX.jaist.ac.jp

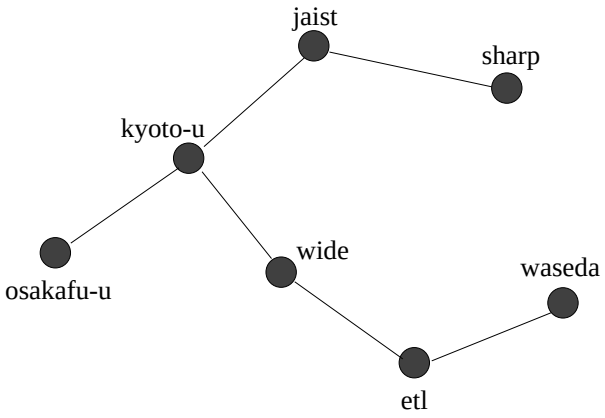


図 15: ロボットサイトで構成した最小木

各PRSが別々に収集した場合のコストの比較

収集対象となるホストとして，日本国内のホストから第2レベルドメインのac, co, go, ne, or の割合を考慮して抽出した約300ホストを使用した．これらのホストを上記のロボットサイトがそれぞれ収集し，7つのロボットサイト(PRS)が共通して収集完了した103ホストについて，コストを比較する．

以下に，それぞれのロボットサイトが，同一の103ホストを収集したときのコストを表2及び図16に示す．

表 2: 103 ホスト収集時のコスト (1999 年 2 月 11 日 (木 / 祝) 午前 2 〜 8 時に収集)

ロボットサイト	コスト
京都大学	604516
早稲田大学	1641665
電子技術総合研究所	2413969
シャープ株式会社	807949
大阪府立大学	1823025
慶應義塾大学	600810
北陸先端科学技術大学院大学	798720

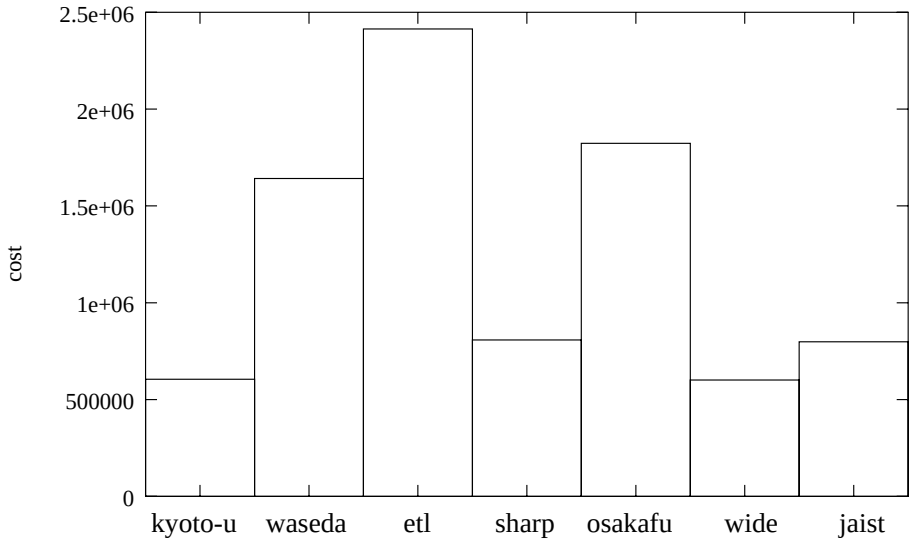


図 16: 103 ホスト収集時のコスト（グラフ）

動作ログをもとにしたランダム分散時の予測値

次に担当ホスト数が同一になるという条件で103ホストを7つのロボットサイトにランダムに割り当てた場合に，先の動作ログから予測したコストを表3，及び図17に示す．このように担当ホスト数を同一にするという条件だけではコストは不均

衡になる．ホストのドキュメント量やロボットサイトからホストまでの距離を考慮していないためである．

表 3: ランダム割り当て時のコスト

ロボットサイト	コスト	担当ホスト数
京都大学	54636	15
早稲田大学	93400	14
電子技術総合研究所	797352	14
シャープ株式会社	224265	15
大阪府立大学	161269	15
慶應義塾大学	70703	15
北陸先端科学技術大学院大学	101899	15

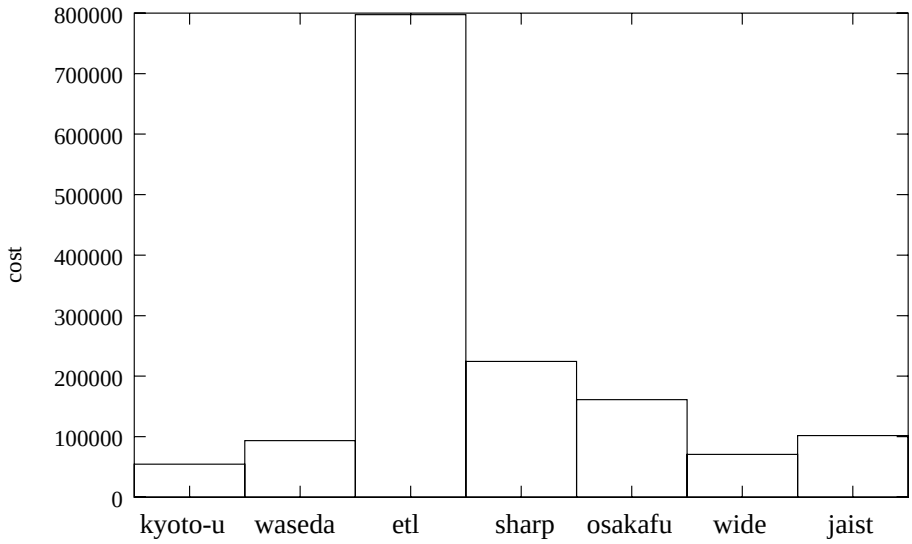


図 17: ランダム割り当て時のコスト（グラフ）

動作ログをもとにした負荷均等化分散時の予測値

このランダム割り当ての状態から先のアルゴリズムを適用し，20 回のコスト均等化を行った場合の「適用回数と各ロボットサイトのコストの遷移」を図 18に示す．この図より，ほぼ5 回程度適用すれば，収束方向へ向かい，20 回の適用で，負荷をほぼ完全に均等化することがわかる．ただし，今回は，7PRS での実験であり，分散する PRS の数が増大した場合には，本アルゴリズムの適用回数は増加するものと考えられる．

ランダム分散時の計算に基づくコストの予測値とそのグラフを表 4と図 19に示す．ランダム割り当て時と比べて収集時間が短縮され，均一化されていることがわかる．

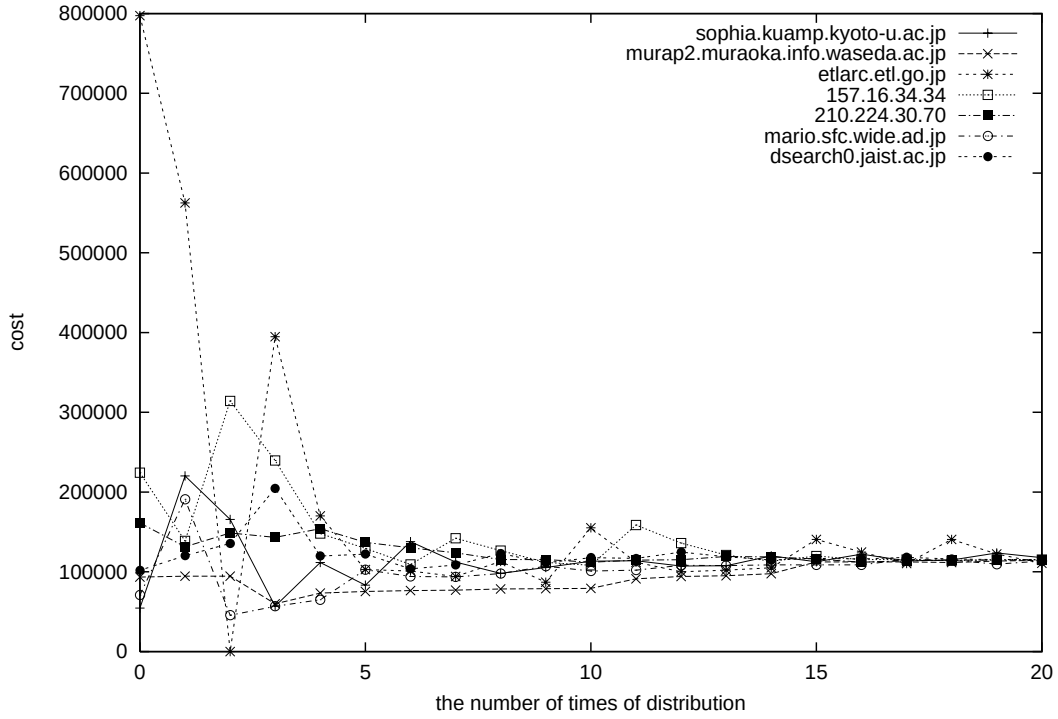


図 18: 均等化アルゴリズムの適用回数と各ロボットサイトのコストの遷移

表 4: アルゴリズムを適用した結果

ロボットサイト	コスト	担当ホスト数
京都大学	117841	32
早稲田大学	114336	29
電子技術総合研究所	110835	6
シャープ株式会社	115412	10
大阪府立大学	115033	7
慶應義塾大学	113526	14
北陸先端科学技術大学院大学	115423	5

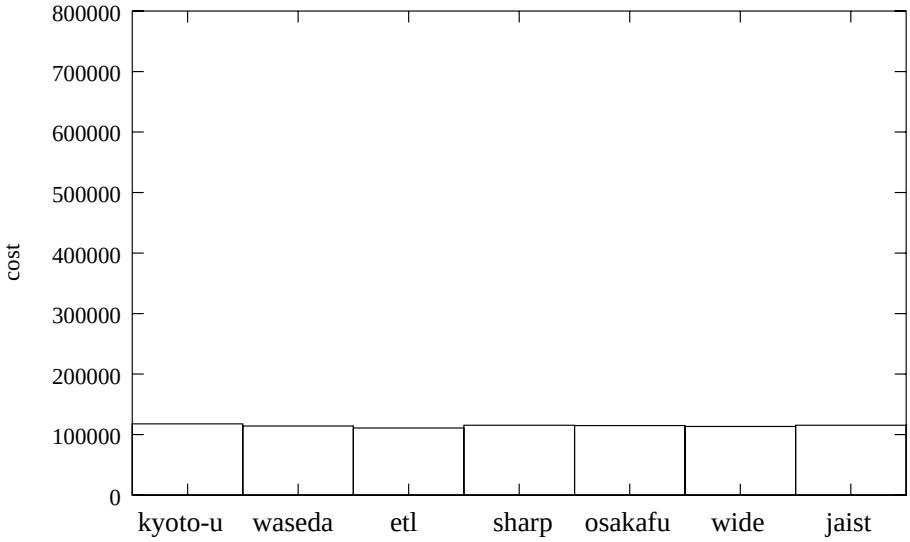


図 19: アルゴリズムを適用した結果（グラフ）

動作ログをもとにした予測値のまとめ

以上の計算に基づく予測から，次のことがわかる.

- ランダムに分散すると，0.76 倍 (慶應) 〜 3.0 倍 (電総研) の高速化が可能．ただし，ランダムな分散によって，京大，慶應の 2 サイトについては，逆に遅くなっるという逆転現象が生じる
- 均等に分散すると，5.1 倍 (慶應) 〜 20.5 倍 (電総研) の高速化が可能.
- 7PRS 程度であれば，負荷均等化のアルゴリズムを 5 回程度適用することにより，収束に向かい，20 回適用すれば計算上の負荷不均衡を 10 秒程度以内にできる.

実験値による比較－ランダム分散時及び均等化分散時

上記で得られたランダム時，及び負荷均等化後の担当ホストのもとで7つのロボット (PRS) を用いて実際に収集を行った際の結果をそれぞれ表5，表6に示す．左が実測値，右が上で得られた予測値である．

表 5: ランダム割り当て時の実測値 (1999 年 2 月 16 日 (火) 午前 2 ～ 8 時に収集)

ロボットサイト	コスト（実測値）	予測値
京都大学	32518	54636
早稲田大学	54620	93400
電子技術総合研究所	113751	797352
シャープ株式会社	227155	161269
大阪府立大学	73800	224265
慶應義塾大学	52542	70703
北陸先端科学技術大学院大学	66283	101899

表 6: コスト均等化後の実測値 (1999 年 2 月 17 日 (火) 午前 2 ～ 8 時に収集)

ロボットサイト	コスト（実測値）	予測値
京都大学	82418	117841
早稲田大学	57680	114336
電子技術総合研究所	13548	110835
シャープ株式会社	51765	115412
大阪府立大学	39630	115033
慶應義塾大学	109936	113526
北陸先端科学技術大学院大学	17885	115423

計算上の予測値と実験値からわかること

以上の計算に基づく予測から，次のことがわかる．

- ランダムに 7 分散すると，2.6 倍 (慶應) ～ 10.6 倍 (電総研) の高速化が可能．
- 均等に 7 分散すると，5.5 倍 (慶應) ～ 22.0 倍 (電総研) の高速化が可能．
- 上記より，ランダムな分散に比べて均等に分散することによって，ほぼ 2 倍の性能向上が得られることがわかる．

さらに，予測値を基準とした時の実測値とコストのずれを表7に示す．

表 7: 実測値と予測値とのずれ

ロボットサイト	ランダム分散時	均等分散時
京都大学	-40%	-30%
早稲田大学	-41%	-50%
電子技術総合研究所	-86%	-88%
シャープ株式会社	+41%	-55%
大阪府立大学	-67%	-66%
慶應義塾大学	-26%	-3%
北陸先端科学技術大学院大学	-35%	-85%

上記に示すように，計算によって算出した値と，実測値の間で，+41% ～ -88% の違いが生じている．以下にこの違いに対して予測される理由と今後，明らかにすべき課題について以下に示す.

① 全体的に予測値より早く収集が終了している点

予測の基準となるデータは，1999 年 2 月 11 日 (木 / 祝) 午前 2 ～ 8 時に収集された．一方，ランダム分散，均等分散時のデータは，同年 2 月 16 日と 17 日の午前 2 時～ 8 時に収集している．つまり，予測の基準となった日は，休日前の夜であり，インターネット利用の負荷が上がり，平日時に比べれば収集に時間がかかったと推測される．そして，この結果，全体的に，ランダム収集と均等分散時の収集時間が短くなったものと考えることができる．実際に，府立大に設置された PRS のデータを解析したところ，URL 数が異なるので正確には判断できないが，表 8のように，約 2.9 倍の収集時間の差があることがわかった.

表 8: 府立大 PRS における同一 47WWW サーバを対象とした収集時間の日別変化

収集日	収集時間	収集 URL 数均等分散時
2/1(月) ～ 2/4(木)	9463152ms	6623URL
2/6(土)	18158930ms	4392URL

② ランダム収集時と均等分散時で予測差の異なるサイト

シャープ及び北陸先端では，ランダム時と均等分散時との間で予測値のハズレ方が大きく異なる．これに対して，他のサイトでは，ランダム分散時と均等分散時の予測値との誤差にあまり違いがない．この原因をログを解析することにより調べた．すると，シャープのランダム収集時の例では，xxx.yyy.kyusyu-u.ac.jp からの収集時間が予測の 10 倍の時間となっており，この特定サイトの収集時間だけでシャープのランダム収集時間の 71% を占めていた．さらに，北陸先端の例でも，ランダム収集時に，特定の xxx.yyy.co.jp からの収集時間が予測の 14 倍の時間となっており，この特定サイトの収集時間だけで北陸先端の

ランダム収集時間の19%を占めていた．また，均等収集時に，特定の1サイトからの収集時間が予測の10分の1の時間で終了していた．

そこで，このような，予測と10倍以上異なっている特定サイトの収集時間が予測通りだったと仮定して，予測値からの誤差を再計算した表を以下に示す．以下に示すように，予測と10倍以上の差があるサイトを除けば，ランダム時と均等時での予測値からの誤差はほぼ同じになる．

表 9: 実測値と予測値とのずれ

ロボットサイト	ランダム分散時	均等分散時
シャープ株式会社	-49%	-55%
北陸先端科学技術大学院大学	-46%	-44%

③ 以上より判明した点

これまでの議論から，「収集時間は曜日による変動が大きい」，すなわち，土日や祝日等の午前2時～8時は，平日の午前2時～8時に比較してネットワークやWWWサーバの負荷が高いことがわかる．また，同じ平日でも，ほぼ収集時間が同じになるWWWサーバと，10倍以上も収集時間が異なるWWWサーバが存在することがわかる．

④ 今後の検討事項１

１～２週間にわたって，収集時間の曜日別の差がどの程度あるかを試験的に調べ，均等分散の基準となるデータを取得する必要がある．

⑤ 今後の検討事項２

同じ平日でも，収集時間に10倍程度以上差がでるサーバとあまり収集時間に変動がないサーバを分類できるかどうかについて検討し，均一分散を行う時に考慮する必要があるかを調べなければならない．

4.3.2 再分配システム

当初の計画通り，収集したデータを配布するWWW ホームページデータ配布サービスの実現可能性についても検討を行った．以下では，まず，再分配システムについて述べる．

分配システムの概説

分配システムは分散ロボットが集めたデータを参加者に分配するシステムである．分配サーバ (distributor) には，次の 3 つの機能が必要である．

① Distributor の PRS からの収集
Distributor の PRS からの収集は，PRS が Distributor に PRS が収集したデータを毎日一定時刻に送信することによって行なわれる．

② Distributor での貯蔵
Distributor では，jp ドメインすべての WWW サーバのデータを保存する．

③ Distributor での分配
Distributor では，ftp プロトコルによって jp ドメインすべての WWW サーバのデータを希望者に対し分配する．

Distributor の PRS からの収集

Distributor の PRS からの収集は，PRS が Distributor に PRS が収集したデータを毎日一定時刻に送信することによって行なわれる．PRS は毎日一定時刻に html ディレクトリを GNUtar+gzip フォーマットにアーカイブし，ftp によって Distributor に PUT する．正当な PRS であることの確認のためこの際 ID とパスワードによる認証を行なっている．また，企業内などで直接インターネットに接続できない PRS の場合は，Delegate Proxy を利用し送信する．

Distributor での貯蔵

各 WWW サーバのデータが PRS ではハッシュのかかったディレクトリ名に保存されているため，Distributor では，各 WWW サーバ毎のに一つの GNUtar+gzip フォーマットのアーカイブし，アーカイブファイルができるように，送信されたデータを分解する．

Distributor からの分配

ftp によって，分解されたデータを公開する．参加者は ftpGET により jp ドメインのデータを取得てきる．なお著作権上の問題があるので，ID と Password による認証を行なう．

各工程間のバッファリング

以上の 3 つの機能の速度差を隠すため，Distributor にはいくつかバッファを用意する．

① Distributor のの収集バッファ

Distributor の PRS からの収集は，PRS が Distributor に PRS が収集したデータを毎日一定時刻に送信することによって行なわれる．PRS からのでーたは数時間をかけてこのバッファに転送される．ftpd は貯蔵機能部にたいしデータの受信イベントを発生できないので，PRS は送信と送信の間に 1 時間以上の間隔をおくことを義務づけられている．

② Distributor での分解前バッファ

受信されたデータは受信の終了後，分解前バッファに移動される．収集バッファと分解前バッファは同一ファイルシステム上にあるためこの移動は一瞬で終る．

③ Distributor での分解後バッファ

分解作業の出力は，分解後バッファに徐々に書き込まれる．

④ Distributor での分配バッファ

分配バッファでは，ftp による公開を行なっているが，更新のため毎日一定時刻 (12:00-13:00) に分解後バッファのデータを分配バッファに移動する．分解後バッファと分配バッファも 同一ファイルシステム上にあるためこの移動は一瞬で終る．この時間帯は分配バッファのデータが不定であるから，参加者はこの時間帯を避けて ftp を行う．

バッファ関の移動処理

バッファ間の移動処理は，同期をとりながら行なわなければならないため一つのプロセスが管理をしている．このプロセスは次の 3 つを順に実行することを繰り返す．

① 収集バッファから分解前バッファへの移動

PRS は送信と送信の間に 1 時間以上の間隔をおくことを義務づけられているので，収集バッファのファイル修正時刻が 10 分以上 1 時間以内ならば，送信が完了したデータなので収集バッファから分解前バッファへ移動する．

② 分解前バッファから分解後バッファへの移動

各 PRS から送信されたデータは，時間をかけて分解し処理後のデータは分解後バッファへ書き込まれる．分解終了後は分解前バッファから消去される．

③ 分解後バッファから分配バッファへの移動

もし時刻が 12:00-13:00 ならば，分解後バッファから分配バッファへの移動を行なう．

性能

300WWW サーバ分の PRS → Distributor への転送を行なった際の性能を記す． Distributor は xxx.yyy.info.waseda.ac.jp であった．

表 10: 主な PRS から Distributor への転送速度

PRS	バイト数	転送秒	転送速度 (byte/sec)
XXX.etl.go.jp	7276922	18	404k
XXX.jaist.ac.jp	6431092	21	306k
XXX.YYY.info.waseda.ac.jp	5509423	99	55k

5 外部発表・新聞発表

5.1 外部発表

5.1.1 ACM SIGIR'98

1998年8月24日～28日に豪国で開催された米国ACM学会主催の「21st International ACM SIGIR Conference on Research and Development in Information Retrieval」において，以下の研究発表(ポスター)を行った.

Hayato YAMANA, Kent TAMURA, Hiroyuki KAWANO, Satoshi KAMEI, Masanori HARADA, Hideki NISHIMURA, Isao ASAI, Hiroyuki KUSUMOTO, Yoichi SHINODA and Yoichi MURAOKA: "*Experiments of Collecting WWW Information using Distributed WWW Robots*", Proc. of SIGIR'98, Melbourne, Australia, pp.379-380, (1998.8.24-28)

SIGIRは，情報検索に関する国際会議の中で最も著名な会議であり，世界24ヶ国から242名の研究者が参加し(米国87名, 豪国77名, 英国13名, 日本12名ほか)，論文発表39件，招待論文1件，ポスター24件，デモ10件の発表が行われた.

招待講演では，米国インフォシーク社のSteve Kirch氏が「インターネット検索の将来」と題して発表し，「実世界では集中制御は稀であり非集中型，すなわち分散型の情報検索が今後重要となる」と結論付けた．これは，本研究開発の方向性が正しいことを示している．ポスターセッションでは，延べ60人の研究者に対し直接説明をすることができ，その中でも，米国IBMやインフォシーク社からは，詳しい内容に入り込んだ質問を受け，関心の高さを認識した.

5.1.2 第17回IPA技術発表会

1998年10月21日～22日に開催された，IPA主催の「第17回IPA技術発表会」において，以下の研究発表(ポスター)を行った.

早大, 京大, 日本IBM, 電総研: Internet広域分散協調サーチロボットの研究開発, IPA第17回技術発表会論文集, pp.222-224 (1998.10.21-22)

5.1.3 GLINT'99

1999年1月26日～28日に開催された，電子技術総合研究所主催の「Symposium on Global Information Processing Technology 1999」において，以下の研究発表(論文)を行った.

Hayato YAMANA: Distributed WWW Search Engines, ,Proc. of GLINT'99, pp.29-35 (1999.1.26-28)

5.1.4 北海道地域ネットワーク協議会シンポジウム'99

1999 年 3 月 4 日〜5 日に開催された北海道地域ネットワーク協議会シンポジウム'99 において，以下の研究発表 (論文) を行った.

山名, 田村, 森, 河野, 黒田, 西村, 浅井, 楠本, 篠田, 村岡: 国内の全 WWW データを 24 時間で収集する分散型 WWW ロボットの試み,North シンポジウム'99 論文集 (1999.3.4-5)

5.2 新聞発表

5.2.1 北國新聞 (1998 年 11 月 14 日朝刊)

1998 年 11 月 14 日の北國新聞朝刊 34 面に「ホームページ新検索ソフト開発へ」という表題で掲載.

6　今後の課題

本年度の研究開発を通じて，目標達成のためには，以下の解決が必要であることがわかった.

6.1　負荷分散単位の細分化の検討

現在の負荷分散単位はWWW サーバ単位であるが，WWW サーバによっては，数万URL のデータを持つものがある．このような大規模サーバに対しては，複数のPRS で協調させ，収集時間の短縮を図る必要がある.

6.2　負荷変動への対応の検討

WWW サーバの能力や負荷によって，収集間隔 (現在は20 秒毎に1URL 取得) を動的に変更する仕組みを構築し，能力の高いサーバに対しては，例えば5 秒に1URL の間隔でアクセスする等の対応を行い高速化を図る必要がある.

さらに，以下の調査を行うことによって，均等分散の基準となるデータの設定方法，及び，サーバの分類が必要であることがわかった.

① １～２週間にわたって，収集時間の曜日別の差がどの程度あるかを試験的に調べ，均等分散の基準となるデータを取得する必要がある.

② 同じ平日でも，収集時間に10 倍程度以上差がでるサーバとあまり収集時間に変動がないサーバを分類できるかどうかについて検討し，均一分散を行う時に考慮する必要があるかを調べなければならない.

6.3　システムの自動バージョンアップシステムの開発

広域に分散した複数のコンピュータを使った実験を行う際には，システムのメンテナンス性を高めることが実験効率化の必須条件である．このため，PRS のソフトウェアを自動的にバージョンアップするシステムの開発が必要である．さらに，今回，全体を管理するサーバPRSM がハードディスクエラーにより3 日間ダウンした．このような事態を避けるためにも，PRSM の二重化も課題である.

7 おわりに

本年度の研究開発では,

① 分散協調サーチロボットのプロトコルの検討

② 分散エリア決定方法の検討

③ 分散協調サーチロボットプロトタイプシステムの設計と先行評価

の 3 点を行った.

7 分散されたプロトタイプシステムを用い，103 の WWW サーバを対象とした収集実験を行い，実際の分散協調サーチロボットの有効性を確認した．具体的には，一カ所で集中して収集する場合に比較し，ランダムな分散をした場合，2.6 〜 10.6 倍の高速化が可能であり，負荷均一化を行った場合，5.5 〜 22 倍の高速化が可能であることがわかった.

今後は，第 6 節で示した課題を解決すると共に，プロトタイプ版の PRS 及び PRSM の信頼性を高め，日本国内に設置された，付録 1 で示される 32 サイトでの分散収集を開始し，当初の目標である 24 時間以内の国内全 WWW サーバデータの収集を目指す予定である.

参考文献

1) AltaVista, http://www.altavista.com/

2) HotBot, http://www.hotbot.com/

3) The Web Robots Pages, http://info.webcrawler.com/mak/projects/robots/robots.html

4) Database of Web Robots, http://info.webcrawler.com/mak/projects/robots/active/html/index.html

5) Netcraft-Web-Server-Survey, http://www.netcraft.co.uk/survey/

6) C. Mic Bowman, Peter B. Danzig, Darren R. Hardy, Udi Manber and Michael F.Schwartz., "*The Harvest Information Discovery and Access System*", Computer Networks and ISDN Systems, Vol.28, pp.119-125 (1995)

7) Network Wizards Internet Domain Survey: http://www.nw.com/zone/WWW/top.html

8) Netcraft Web Server Survey: http://www.netcraft.co.uk/survey/

9) Steve Lawrence, C. Lee Gilesd: *Searching the World Wide Web*,Vol,280, No.5360, Issue 3, pp. 98 - 100 (1998.4) (http://intl.sciencemag.org/)

10) CERN: http://www.cern.ch/

11) NCSA Mosaic Home Page: http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/

12) 山名早人: *WWW*情報検索の現状, コンピュータソフトウェア, Vol.14, No.5, pp.67–74 (1997)

13) Goo:http://www.goo.ne.jp/

14) Yahoo!: http://www.yahoo.com/

15) WWW 情報検索サービスの動向: http://www.etl.go.jp/~yamana/Research/WWW/survey.html

16) MetaCrawler: http://www.metacrawler.com/

17) Infoseek: http://www.infoseek.com/

18) 千里眼: http://senrigan.ascii.co.jp/

19) Internet Research Task Force (IRTF): http://www.isi.edu/irtf/irtf.html

20) Defense Advanced Research Projects Agency(DARPA): http://www.darpa.mil/

21) National Science Foundation(NSF): http://www.nsf.gov

22) WebAnts: http://polarbear.eng.pgh.lycos.com/webants/

23) –: *WWW* サーチエンジンの未来, Conference Notes, Networld+Interop 98 Tokyo, pp.1-22 (1998.6.3)

24) P.Francis, 神林隆, 佐藤進也, 清水奨: “次世代情報検索インフラストラクチャ *Ingrid*”, NTT R & D, Vol.45, No.2, pp.159-165 (1996) (http://www.ingrid.org/papers/RDpaper.html)

25) 森 英雄: 分散協調型Web データ収集アルゴリズムの性能評価, 京都大学 情報学科 特別研究報告書 (1999.2)

26) 西田竹志:“～ *LAN* 間接続～ *TCP/IP* インターネットワーキング”, ソフト・リサーチ・センター (1993)

27) 茨木 俊秀: “アルゴリズムとデータ構造”, 昭晃堂 (1989)

付録 1

1999 年 *2* 月 *20* 日現在の実験協力サイト一覧 *(*順不同*)*

1. 東京大学 工学部 電子情報工学科
2. 学術情報センター研究開発部
3. 北陸先端科学技術大学院大学
4. 奈良先端科学技術大学院大学情報科学研究科
5. 東京大学 生産技術研究所 第 3 部
6. 岐阜大学工学部応用情報学科
7. 農林水産省農業研究センター
8. シャープ株式会社技術本部ソフトウェア研究所
9. 京都大学大学院情報学研究科
10. 電子技術総合研究所情報アーキテクチャ部
11. 慶應義塾大学環境情報学部
12. 関西大学　工学部　電気工学科
13. 徳島大学工学部
14. 岩手県立大学ソフトウェア情報学部
15. 東京工業大学 大学院総合理工学研究科
16. 横浜国立大学工学部電子情報工学科
17. NTT 光ネットワークシステム研究所
18. (株) デオデオインターネットサービス
19. 琉球大学 総合情報処理センター
20. 南山大学経営学部情報管理学科
21. 大阪府立大学工学部経営工学科
22. (株) ネットウェーブ四国　技術部
23. トライネットワークインターナショナル（株）
24. 北海道地域ネットワーク協議会
25. ネスク・インターネット (株式会社日本海ネット)
26. 東北インターネット協議会
27. 早稲田大学理工学部情報学科
28. 日本 IBM 東京基礎研究所
29. Intio プロバイダ
30. 九州大学大学院システム情報科学研究科
31. 広島大学工学部第二類
32. 東海地域ハブ研究会