2R-6

分散処理における Web サイト評価システム KAGAMI の拡張

竹内 康夫 † 平石 広典 †† 溝口 文雄 †

† 東京理科大学理工学部

†† 東京理科大学総合研究所

1 序論

近年，企業等の Web サイトは本来社員らが行う総務や広報などの業務機能を備えるなど重要な役割を担っている．従って，それらを評価・診断することは有効といえる [3]．そうした中，人間による評価で生じる主観的な判定を排除し，HTML データをコンピュータで解析することにより、客観的な判定を下す Web サイト評価システム KAGAMI[1] があるが，Web サイトの巨大化に伴い処理時間を多く必要とする問題点がある．本稿では KAGAMI において処理時間のボトルネックであった，ネットワークを通じた HTML ファイルのダウンロードを担う処理をサイトのツリー構造に着目した分散を行い処理時間の高速化を図る．

2 Web サイト評価システム KAGAMI

Web サイト評価システム KAGAMI はネットワークを通じて評価対象の Web サイトの HTML ファイルをダウンロードし，構文解析することによりサイトの概観 [2]，情報分布，内外へのアクセス度などの評価を出力するものである．図 1 にその出力例を示す.KAGAMI は全て Java 言語で実装されている．これによりプラットフォームを選ばない実行が可能である．

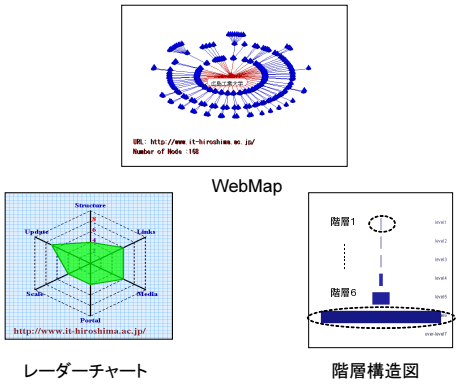

図 1: KAGAMI における評価結果の出力

2.1 データ構造

本節では，KAGAMI に必要なデータを格納する Node 及び NodeList のデータ構造を示し，その役割について述べる．

Node とは 1 つの HTML ファイルに対応して作成されるデータの格納先であり，この Node の HTML ソースを文字列として格納し，親 Node 及びリンク先の HTML を自身の子とし, それに対しても Node を作成し子 Node のリストを作成し格納する．また，複数の Node の統合先として NodeList を作成した．NodeList には最終的に Web サイトを表現するデータが全て格納される．NodeList 内の open には各 Node の未処理の子ノードが格納され，ここからタスクが配分される．(図 2)．

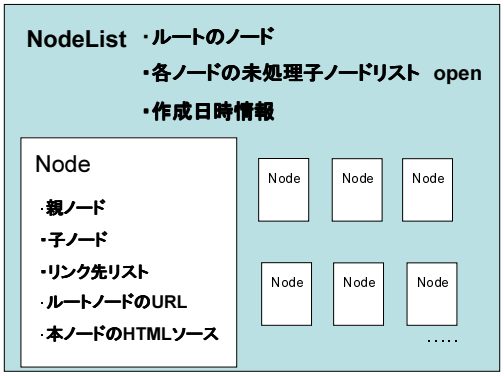

図 2: Node および NodeList のデータ構造

2.2 基本処理

KAGAMI の基本処理は大きく HTML ダウンロードとパージングにわかれる．本論で述べるパージングとは HTML ファイルを構文解析により HTML タグを抽出し，リンク情報や評価に要するパラメータを算出することをいう．

先ず，Node に格納された HTML ファイルをパージングすることによりリンク情報を得る．各リンク先ごとに Node を作成し URL を格納後、HTML をダウンロードし格納する．これらの Node を自身が親となる子 Node と呼ぶ．次に処理後の Node を NodeList に格納し，子 Node 群を NodeList の open に格納する．最後に open に格納された Node の処理を開始する．これらを繰り返すことによって Web サイト全体を評価することが可能になる．

3 処理方式

3.1 逐次処理方式

逐次処理方式は既存の KAGAMI が採用している方式で，NodeList の open に入ったリンク先のノードか

Web site evaluation system KAGAMI enhancing in decentralized processing

Yasuo Takeuchi†, Hironori Hiraishi††, Humio Mizoguchi†

{†Faculty of Sci.and Tech,Tokyo University of Science, ††Research Institute for Science and Technology}

ら順にネットワークを通じてデータを取得する処理を実行する．処理要求を受けた URL の HTML をダウンロードし，パージングし，リンク先の子 Node の URL を得る．子ノード群に対し現在結果が格納されている NodeList の中の結果 Node 及び open に格納されている処理待ちの Node と重複がないかチェックする（以後重複チェック）．この処理により不要なリンクのダウンロードを避けることができる．

3.2 マルチスレッド方式

マルチスレッドとは処理単位に対し CPU の処理時間の割り当てを細かく分割することで並列処理を実行する手法である．本論では，ネットワーク取得処理においてマルチスレッドを適用する．任意のページの Node を処理する際，NodeList の open に格納されたリンク全てに対してスレッドを同時に実行する．これによって、リンク切れなどによるタイムアウト処理などで処理が滞っているノードにより，他のノードの処理が実行されない事態を防ぐことができる．各スレッドは生成時に NodeList を渡しており，子ノードのダウロード前に重複チェックを行うことで不要なリンクのダウンロードを防ぐことができる．しかし，同時実行中の別スレッドの処理は反映されない為完全ではない．これは次節での GRID 手法にも同様のことが言える．

3.3 GRID 方式

本論で扱う GRID 方式とは，マスタと複数のワーカがネットワークを介して接続する環境下でマスタがタスクをワーカに分配し，その結果をマスタで統合することによって，最終的な処理時間を短縮する方式である．本論では前節で述べたマルチスレッド方式で NodeList の open に格納された Node ごとに生成したスレッドを生成したのと同様に，そのスレッドに任せた処理をワーカマシンに委ねる形式をとる．ワーカが open に格納された Node とその時点での NodeList を投げワーカが基本処理後マスタに戻す．マスタでは結果を NodeList に格納し，結果の子ノードを重複チェック後 open に格納する．これら一連の流れを繰り返すことにより全体のデータ取得及び評価を行う．

4 評価実験

4.1 実験

前章で述べた逐次，マルチスレッド，GRID の各手法を東京理科大学，明治大学，法政大学の Web サイトで検証を行う．逐次，マルチスレッド，GRID のマスタについては Pentium4 2.8GHz，メモリ 1 GB のマシン．ワーカマシンに Celelon2.8GHz2 台，2.4GHz2 台，メモリ 256MB のマシンを用いて処理時間の検証を行った．

4.2 結果及び評価

図 3 に各サイトと処理時間の関係，表 1 に総リンク数とリンク切れの数の関係を示す．

図 3，表 1 よりリンク切れ率が高い東京理科大学及び明治大学の Web サイトは逐次処理に比べてマルチスレッド方式が効果が高く，逆に低い法政大学の Web サイトでは逐次処理がマルチスレッドの処理時間を上回る結果を得た．これは逐次処理ではリンク切れページ遭遇時のタイムアウトに時間を要し，その間次の処理が滞ることが原因であることが考えられ，マルチスレッドでは滞っている処理と同時に別スレッドが進行している為、それらの問題を回避できていると考えることができる．4 台のマシンで分散した GRID 方式は台数効果を得ることはできず，全ての Web サイトでマルチスレッド方式より台数効果を得ることはできなかった．

表 1: 総リンク数とリンク切れの関係

Web サイト	総リンク	リンク切れ	割合
東京理科大学	1431	1114	0.44
明治大学	9633	4635	0.48
法政大学	4995	575	0.12

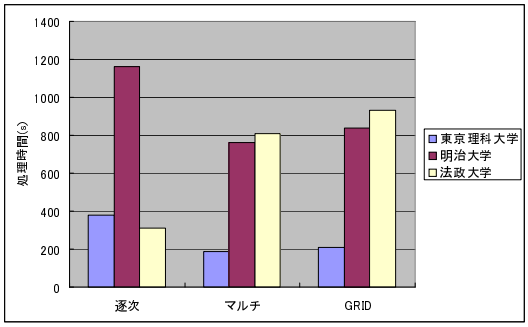

図 3: 各 Web サイトと処理時間の関係

5 結論

本論では Web 評価システム KAGAMI に対しマルチスレッド方式，GRID 方式を導入することにより処理時間の短縮を図った．マルチスレッド方式ではリンク切れなどによる処理停滞を回避する効果が得られ，処理時間の短縮を実現した．しかし GRID 方式では大容量の通信データによる通信コストにより台数効果を得ることは出来なかった．今後の課題としてはワーカー間でのタスクの協調などより効率のよいタスクの配分を導入や通信データの改善を行い台数効果を上げていく必要がある．また，Web 評価システム KAGAMI だけでなく，Web サイトデータを利用する他のシステム，アプリケーションに対して適用できる汎用性を持たせていく必要がある．

参考文献

[1] 大塚尚典，平石広典，溝口文雄，“ KAGAMI:リンク構造に基づいた Web 格付けエージェント“ ，人工知能学会全国大会 15 回論文集，2001.5.23.

[2] 沢井宏，大和田勇人，溝口文雄，“ WebMap:Hyperboric Tree を利用した WWW ブラウジングの支援 ”, 情報処理学会第 58 回全国大会講演論文集,pp.3-67〜3-68,1999.3.10

[3] アットアス・コーポレーション&編集部, 上場企業ホームページ総攬, 光文社,2003.