NAP-Web における次回アクセスを支援するクライアント機能

最所研究室 08G460 富田貴仁

Web システム「NAP-Web」は，サービスが継続できないとき，整理券を用いて負荷を時間的に平滑化する．NAP-Web は，整理券を持ったアクセスを確実に受け付けるために余裕を持たせているが，この余裕によって処理効率の低下が生じる．本研究では，この効率低下を防ぐためのクライアント・サーバ間での連携を行うクライアント機能の開発を行う．クライアント機能は，サーバの状況を確認するためにポーリングを行い，サーバに余裕があれば整理券に書かれている時刻よりも早くアクセスする．本稿では，クライアント機能の開発とポーリング方法の評価について述べる．

1. はじめに

近年インターネットでは，Web サーバへの負荷が増大し，リクエストの急増により，いつサービスが受けられるのか分からない状態になることがある．この問題に対処するため，我々の研究室では，整理券を用いた Web システム「NAP-Web」の開発を行っている[1]．NAP-Web は，アクセスが集中した際，一部のリクエストを拒否するが，代わりに次回アクセスのための整理券を配布する．そして，拒否されたクライアントが，整理券の書かれた時刻に再度リクエストを行った場合にサービスをすることを保証する．このようにして，「次回いつになればサービスを利用できるのか」というユーザの不満を抑えることができる．しかし，リクエストの種類やシステムの状況により，サービスできる時間が大きく変動するので，次回アクセス可能時刻を正確に予測することは困難である．そこで，NAP-Web では次回アクセスが可能となる時刻を予測する際に，失敗しないように余裕を持たせている．しかし，図 1 に示すように，整理券が配布されている待機中のリクエストが存在するにもかかわらず，サーバの手が空いているという状況が発生してしまう．

本研究は，サーバとクライアントの連携により，NAP-Web における上記の問題を解決することを目的とする．具体的には，クライアント側からサーバの状況を確認し，サーバが処理可能であるならば，整理券に書かれている時刻になっていなくても自動でアクセスを行うことにより，次回アクセスを効率化するシステムを開発する．このクライアント側で動作するシステムのことをクライアント・サーバ (以下 CS) 連携モジュールと呼ぶ．CS 連携モジュールは，サーバの状況を確認するためにポーリングを行う．このポーリングの間隔が短いとサーバへの負荷を増やすことになり，処理の遅延を招く可能性がある．逆に間隔が長いとサーバの空き時間を十分には解消できず，効果の薄いものとなってしまう．そのため，少ないポーリング回数で，最適なアクセスタイミングを得る方法が必要である．このアクセスタイミングを決定するためのアルゴリズムを監視アルゴリズムと呼ぶ．監視アルゴリズムは，サーバから得られる処理の統計情報を参考にして待ち時間の算出を行う．この監視アルゴリズムを策定し，実験を行い，よりよいタイミングのアクセスを行う方策を検討した．

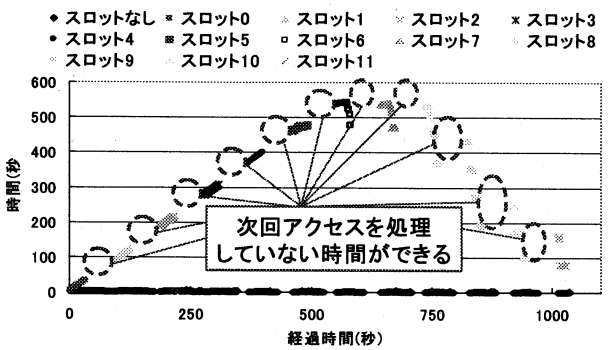

図1　サーバの処理の空き時間の例

2. CS 連携モジュールの動作

CS 連携モジュールは，クライアントでサーバの処理状況を監視する．図 2 に示すように，サーバの処理が予定より早く終わった場合は，予定を繰り上げて自動的に Web ページを取得する．

このモジュールは図3に示すオブジェクト群により構成される．サーバが過負荷状態に陥っている場合，CS 連携モジュールが呼び出され，以下の動作を行う．

・チケットレシーバでブラウザから整理券を受け取り，チケット解析が内部表現であるチケットブジェクトに変換する．

・コネクションコントローラは，コネクトタイマにチケットオブジェクトを渡し，サーバ監視の時刻まで待ち，

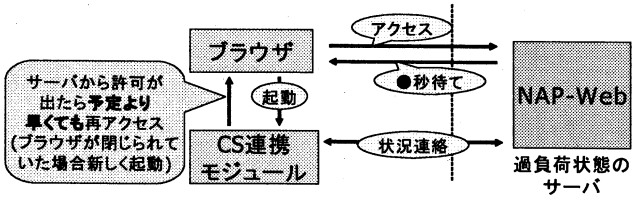

図2　CS 連携モジュールの動作の概要

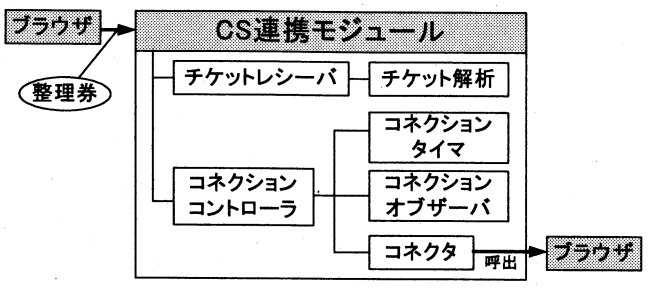

図3　CS 連携モジュールの構成

通知するよう指示する.
・コネクトタイマから通知されたらコネクションオブサーバでサーバ状況を確認し，コネクタでWebページへのアクセスを行う.

3. サーバ状況の確認

NAP-Webは，スロットと呼ぶ概念を用いて整理券開始時刻と終了時刻を算出する.「スロットとは，次回アクセスを可能とする時間を一定の間隔で区切ったものである.例えば，1スロットの期間を10分間とすると，サーバの処理状態から10分間で処理可能なアクセスの数を算出し，その数を1スロットで発行できる整理券の最大枚数とする.同一スロットで発行された整理券を持つ次回アクセスを同一グループとして扱う.ここで余裕を持たせるためにスロット期間に整理券の有効期間分を加えている.もともとのスロット期間をスロット有効期間と呼ぶことにする.処理予定のスロット番号 (有効スロット番号) と実際に処理中のスロット番号 (処理スロット番号) を比べることにより，サーバでの処理状況を判別できる.

4. サーバの処理状況の監視アルゴリズム

ポーリングのタイミングを決定するためのアルゴリズムとして以下の3つを考えた.

方法1：静的なタイミングの決定

アクセス開始時刻までの1/2などの時間に1回目のポーリングを行う.この時点で，まだアクセスが許可されていない場合，同様の時間 (ここでは1/2) が経った時に次のポーリングを行う.アクセスの許可が出るまでこの動作を繰り返す.

方法2：スロット番号を用いて動的に決定

処理スロット番号と有効スロット番号の差と，スロット有効期間の積の分だけ早めにポーリングする.

方法3：スロット番号と統計情報を用いて動的に決定

整理券のスロット番号と処理中のスロット番号の差と，スロット間隔と実際にスロットの処理にかかった時間の差 (短縮時間) の積の分だけ早めにポーリングする.

5. 実験と評価

多数のクライアントが，CS連携モジュールの機能を用いて同時にサーバを監視するシミュレータを作成した.紙面の制限で，スレッド数400，1スレッドにおけるリクエスト回数は50回，1回のリクエストで1MBの画像ファイルを1つ取得する実験の結果のみ示す.スレッド数は，一斉にWebサーバにアクセスするクライアントの台数にあたる.方法1，方法2，方法3の実験結果を表1に示す.示された待ち時間，実際の待ち時間，およびDL(ダウンロード) 時間は1リクエストあたりのそれぞれの平均値であり，時間の単位は全て秒である.方法2では，方法1に比べ待ち時間は少し改善されたが，整理券の配布枚数や終了時間などは少し増加した.また，サーバの処理の空き時間も少し増加していた.方法3では，方法2に比べ待ち時間の平均値はさらに良い結果となったが，各リクエストのアクセスタイミングに大きなばらつきが発生し，その他の項目では悪化してしまった.これは，前後のスロットでアクセスのタイミングが重なり，アクセスの度合いがうまく平滑化できなかったことが原因であると考えられる.

これを考慮し，方法2を元に改良を行うことにした.具体的には，スロット有効間隔の替わりに，スロット有効期間と短縮時間の和を定数で割った値を補正値として用いる.定数値を変え実験を行い，その中で最も結果の良かったものを表2に示す.クライアントに示される待ち時間と，実際の待ち時間は静的な方法より，動的な方法の方が若干良い結果が得られた.しかし，終了時間は改良前より若干よくなったが，静的な方法に追いつくことはなかった.しかも，整理券発行枚数は増加し，処理の空き時間も改善しなかった.その原因を調べたところ，有効スロット番号や処理スロット番号が1つ進むごとに監視時間に間が空いてしまうことであることが分かった.動的な方法についてさらに改良を行った場合，待ち時間を減らすことはできても，サーバの処理の空き時間を減らすことは困難である.現状でのスロット番号の情報だけでは，これ以上の改良は望めない.

表1 3つのアルゴリズムの実験結果

	示された待ち時間	実際の待ち時間	DL時間	整理券配布枚数	終了時間
方法1	375.2	188.5	1.886	2578	1809
方法2	319.0	182.5	1.960	2760	1887
方法3	301.2	159.3	1.975	3075	1957

表2 改良したアルゴリズムでの実験結果

	示された待ち時間	実際の待ち時間	DL時間	整理券配布枚数	終了時間
改良	251.0	160.0	2.020	2969	1861

6. おわりに

NAP-Webの次回アクセス補助機能として，クライアントとサーバの連携モジュールを開発した.監視アルゴリズムの評価のため，CS連携モジュールの機能を用いた負荷クライアントの作成し，実験を行った.その結果，静的な方法の場合に比べ，待ち時間を多少減らすことはできたが，どちらの方法もサーバの処理に空きができしまうことを完全には解決することができなかった.現状ではサーバの処理の空き時間を埋めることができないため，サーバの処理の進み具合を具体的に知るための情報を，もっと詳細に通知してもらう必要がある.今後の課題として，CS連携モジュールを実際に導入した際の詳細設定など細かい仕様を考え，サーバから得られる情報が詳細になった場合の実験を行う必要がある.また，CS連携モジュールのフレームワーク化についても検討する.

【参考文献】

[1] 加地智彦，"急激な需要増加による負荷を整理券により時間的に平滑化するWebシステムの開発 "，香川大学工学部2009年度博士論文(2010).