プロキシサーバによる効率的なWeb閲覧履歴の取得に関する研究

岡山大学 大学院自然科学研究科
栗原 聖治

研究背景

利用傾向の把握のために利用者のWeb閲覧履歴を収集

利用者が本当に見ていたWebページのURLのみ取り出したい

<利用者が本当に見ていたWebページのURL >

(1) Webページの階層構造のトップのURL(MainURL)

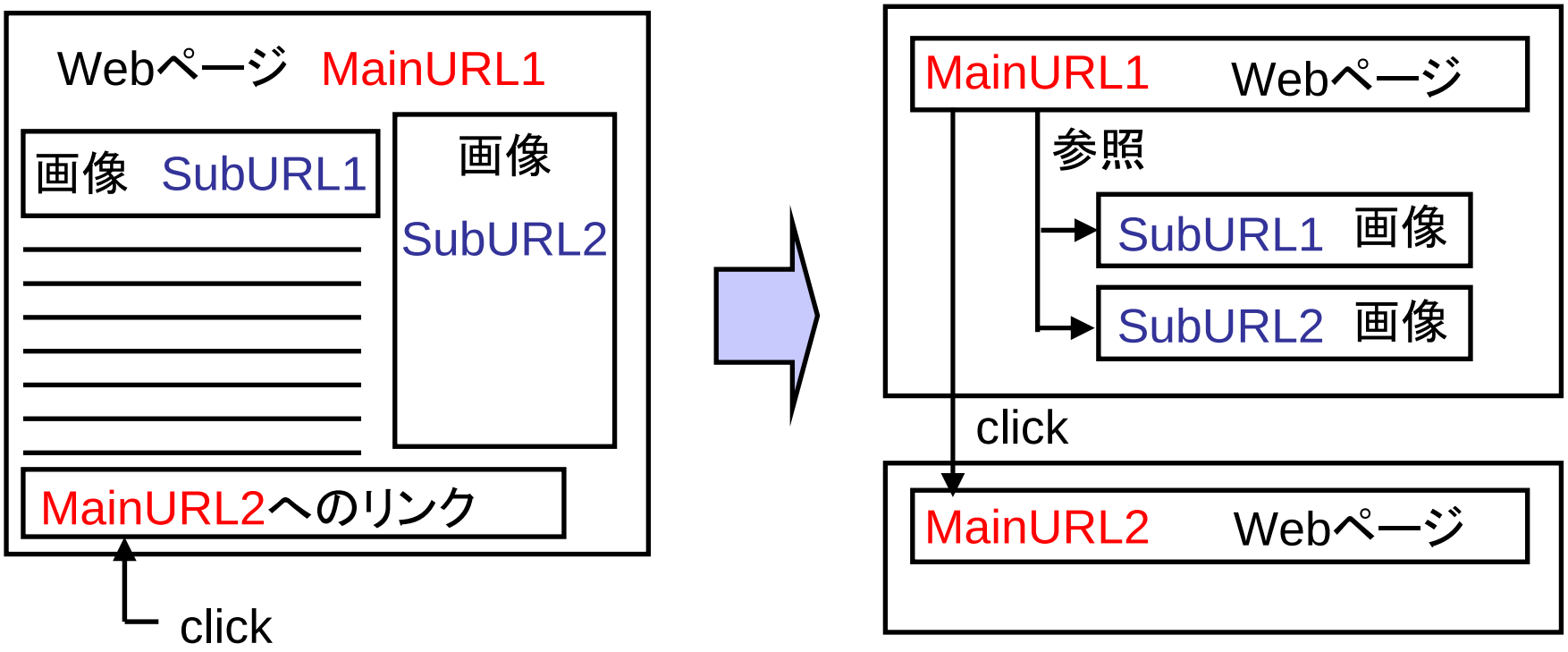
(2) 利用者が時間をかけて閲覧したページのURL

必要な情報のみを効率的に抽出するフィルタが必要

MainURLを判別するため，Webページの構造に着目

Webページの構造

一般的なWebページは複数の構成要素から成る



MainURL：Webページの階層構造トップのURL
SubURL：MainURLに付属するURL

Webページの構造に着目したフィルタ

フィルタリングの条件を以下のように設定

(条件1) リファラになっている or リファラがない履歴

リファラ：履歴内でURLの参照関係を示す情報

(条件2) サービス状態コードが200(OK) or 304(Not Modified)

(条件3) Content-Typeがtext/htmlの履歴

⇒ CSSやXMLなどの誤ったデータを除外

(条件4) 前の候補ページを参照してから一定時間(A)経過

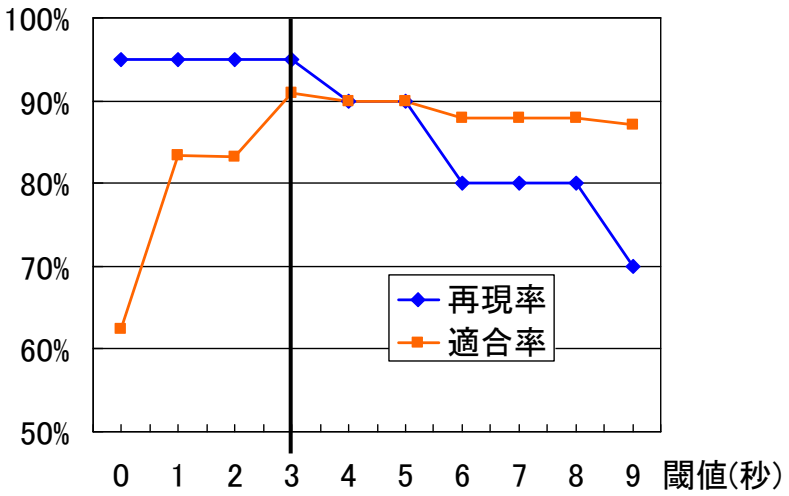
⇒ ブラウザがMainURLからSubURLを参照する間隔は，利用者がMainURLから遷移する間隔より充分短い

(条件5) 後ろの候補ページとの参照時間差が一定時間(B)以上

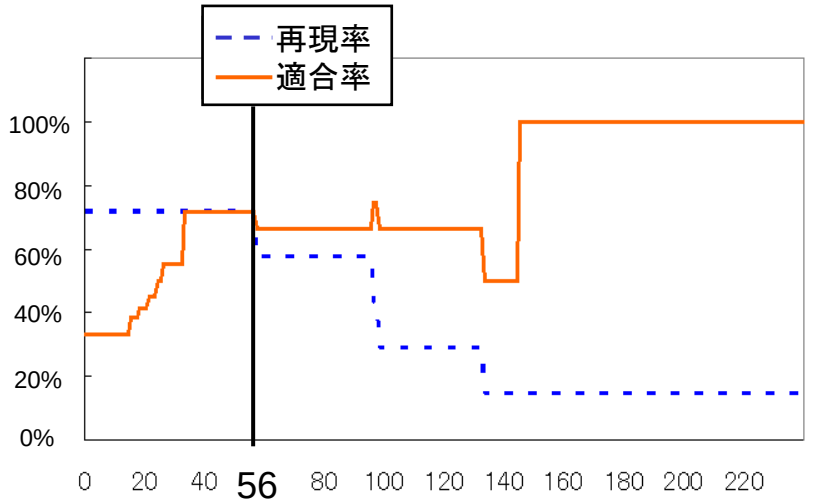
⇓

一定時間(A)を機械参照閾値，一定時間(B)を閲覧閾値として設定

機械参照閾値と閲覧閾値の設定



機械参照閾値の評価



閲覧閾値の評価

再現率と適合率はトレードオフの関係

再現率を保ちながら，適合率を向上できる閾値を設定

機械参照閾値を3秒，閲覧閾値を56秒に設定

課題

(課題1) 複数ウィンドウでの閲覧を考慮していない

⇒ 新規ウィンドウで閲覧候補を次々に開くとMainURLを短い間隔で参照するため正解データを見逃す

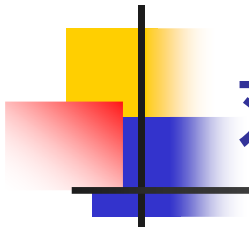
(課題2) 閾値や保存するデータは利用者に大きく依存する

⇒ 人や仕事にあった閾値やContent-Typeにカスタマイズできるほうがよい

(課題3) フィルタリング後のデータに利用者が手を加えられない

(課題4) Webブラウジングを一面的にしか扱えない

⇒ 2つ以上の仕事を同時に行った際に，判別できない



対処

(対処1) Webブラウジングの流れを木構造として推測

通信履歴の参照関係をより詳細に扱い，課題1に対処

(対処2) パーソナライズ機能の追加

抽出に関する設定を利用者が変更可能にし，課題2に対処

(対処3) 利用者によるデータの操作機能の追加

フィルタリング後のデータを見やすく提示し，利用者がデータの扱いを操作できるようにし，課題3に対処

(対処4) Webブラウジングの流れから仕事を推測

異なる仕事は異なる木で行われると推測し，課題4に対処

以降，対処1について述べる

ブラウジングツリーの構成

リファラの参照関係からブラウジングツリーを構成

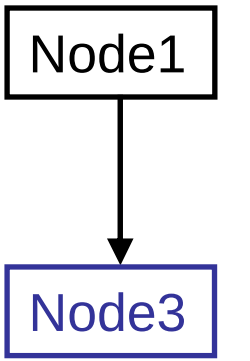
(A) 各Nodeは参照履歴1つに相当

(B) 木の根はリファラがない通信履歴

(C) 各Nodeの関係はリファラによる親子関係

NodeN：MainURLのノード，Nは参照順番

→：親子関係



Node2

Node1

Node2

リファラがないNode

Node3：リファラがNode1のNode

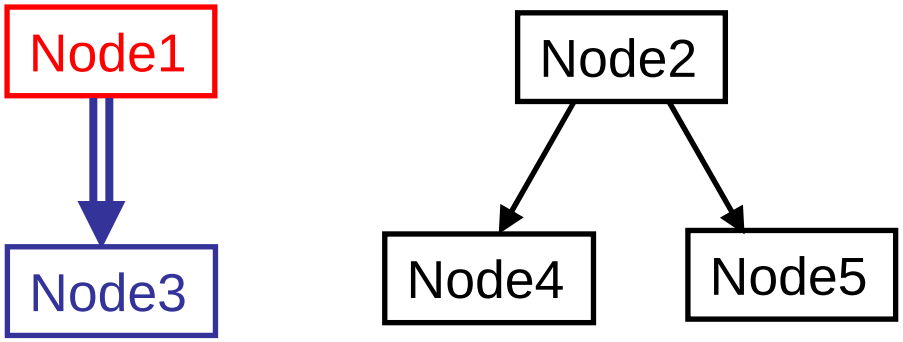
閲覧時間の推測

MainURLのNode同士で閲覧時間を推測

(1) 利用者が直接クリックによって遷移したNode

⇒ 自身の子のNodeとの参照時刻差

Node3への遷移があったNode1の閲覧時間



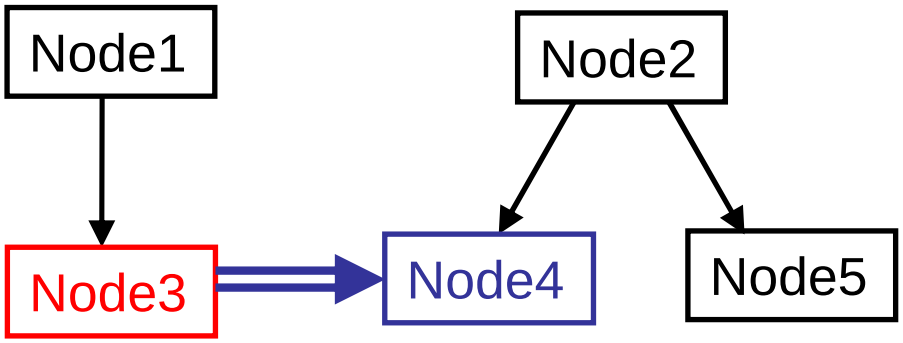
Node1の閲覧時間 ＝ Node3の参照時刻 － Node1の参照時刻

直接クリックによらない閲覧時間の推測

(2) 利用者が直接クリックによって遷移しなかったNode

⇨ 子がないので，次に追加されたNodeとの参照時刻差

遷移のなかったNode3の閲覧時間



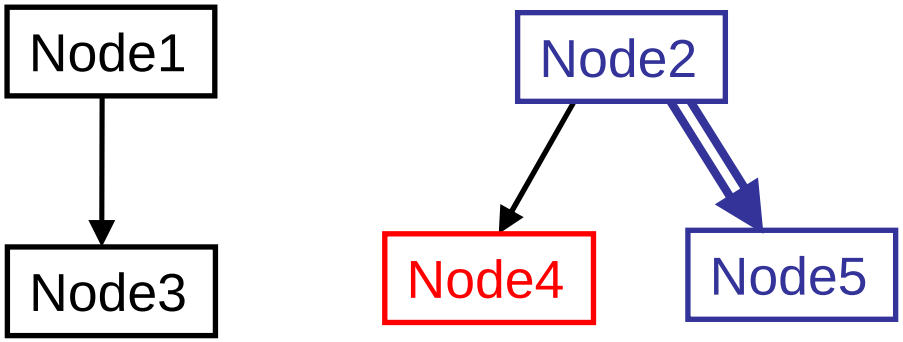
Node3の閲覧時間 ＝ Node4の参照時刻 － Node3の参照時刻

複数ウィンドウによる閲覧時間の推測

(3) 自身の次に追加されたNodeと親Nodeが同じ場合

⇒ 次に追加されたNodeと親Nodeとの参照時刻差

次のNode5が同じ親Node2から遷移したNode4の閲覧時間



Node4の閲覧時間 ＝ Node5の参照時刻 － Node2の参照時刻

(特徴1) 閲覧時間が，改良前に比べて長く算出される

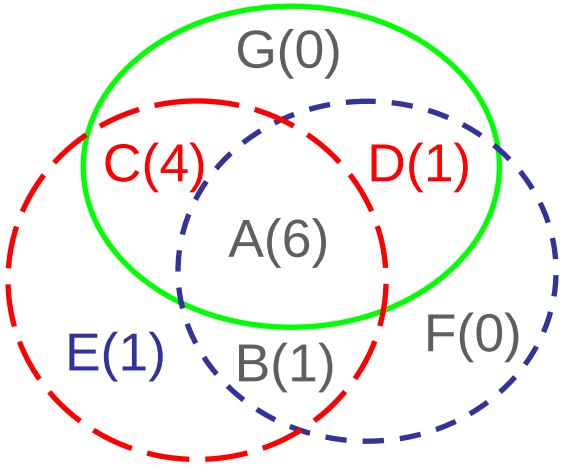
(特徴2) 単一ウィンドウによる閲覧で，同様の通信履歴になった場合，閲覧時間が実際より長く算出される

⇒ 再現率が向上する

評価

全データ数248個，正解データ数11個のデータに対して，機械参照閾値3秒，閲覧閾値56秒でフィルタリングし比較

フィルタ	抽出データ数	抽出正解データ数	再現率	適合率
改良前のフィルタ	A+B+D+F=8個	A+D=7個	64%	88%
改良後のフィルタ	A+B+C+E=13個	A+C=10個	91%	77%



＝正解データ

＝改良前のフィルタ

＝改良後のフィルタ

一方のフィルタがもう一方を包含していない

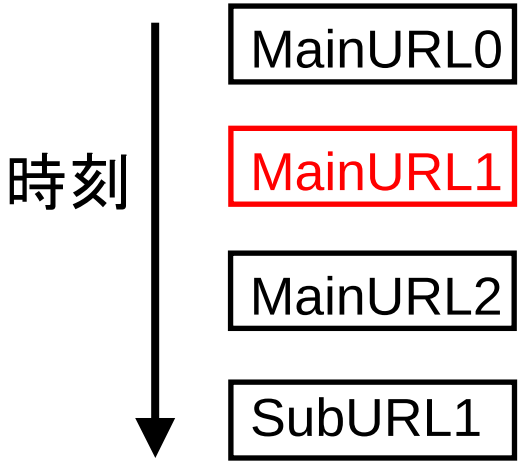
⇨ お互いに取り逃した正解データが存在

考察

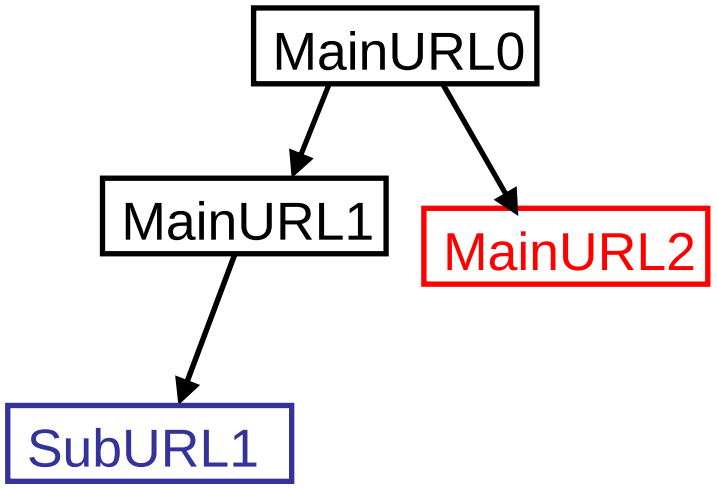
一方のフィルタがもう一方を包含していない

⇒ お互いに取り逃した正解データが存在

<改良前のフィルタ>



<改良後のフィルタ>

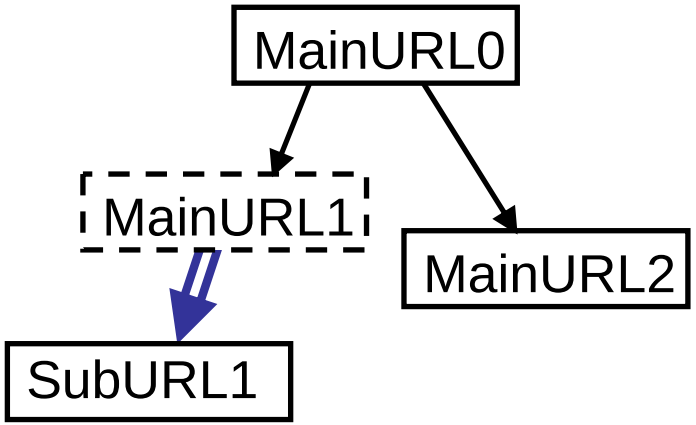


: どちらか一方でのみ抽出した正解データ

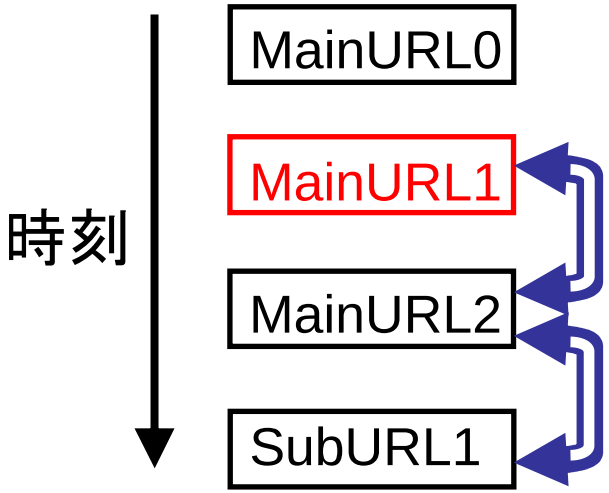
: 流れに着目した場合に誤って抽出したデータ

MainURL1に関する考察

MainURL1：改良前のフィルタでのみ抽出した正解データ



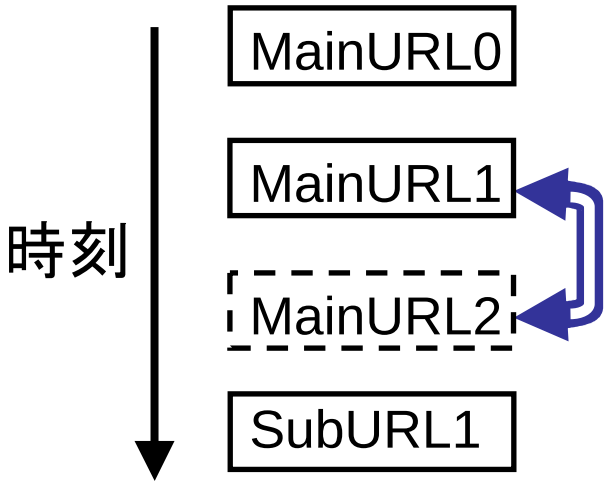
機械参照閾値より長いためSubURL1をMainURLだと判断
MainURLと推測したが，閲覧時間が閲覧閾値より短いため排除



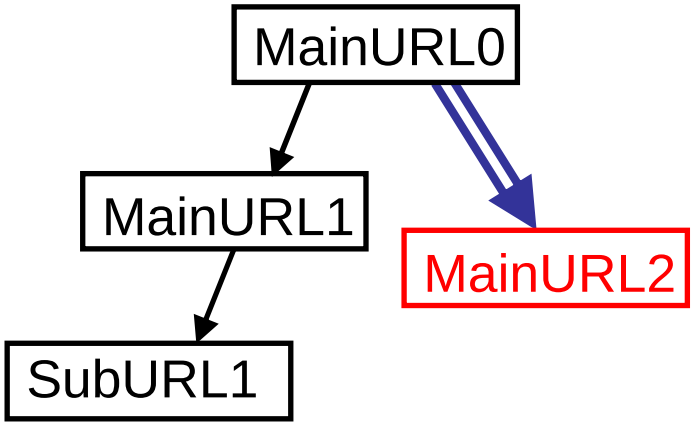
機械参照閾値より短いためMainURL2とSubURL1をSubURLと判断
MainURLと推測し，閲覧時間が閲覧閾値以上であったため抽出

MainURL2に関する考察

MainURL2：改良後のフィルタでのみ抽出した正解データ



直前の履歴との参照時刻差が機械参照閾値より短い
SubURLと推測し，排除



MainURL0との差なので機械参照閾値より長い
MainURLと推測し，閲覧時間が閲覧閾値より長いため抽出

本発表のまとめ

<プロキシサーバによる効率的なWeb閲覧履歴の取得に関する研究>

(1) フィルタの設計方針

(A) 利用者が本当に見ていたURLのみの取得

(2) 実現のための課題と対処

(3) ブラウジングツリー

(A) ブラウジングツリーの構成

(B) ブラウジングツリーの評価

<今後の課題>

(1) 妥当な閾値の調査

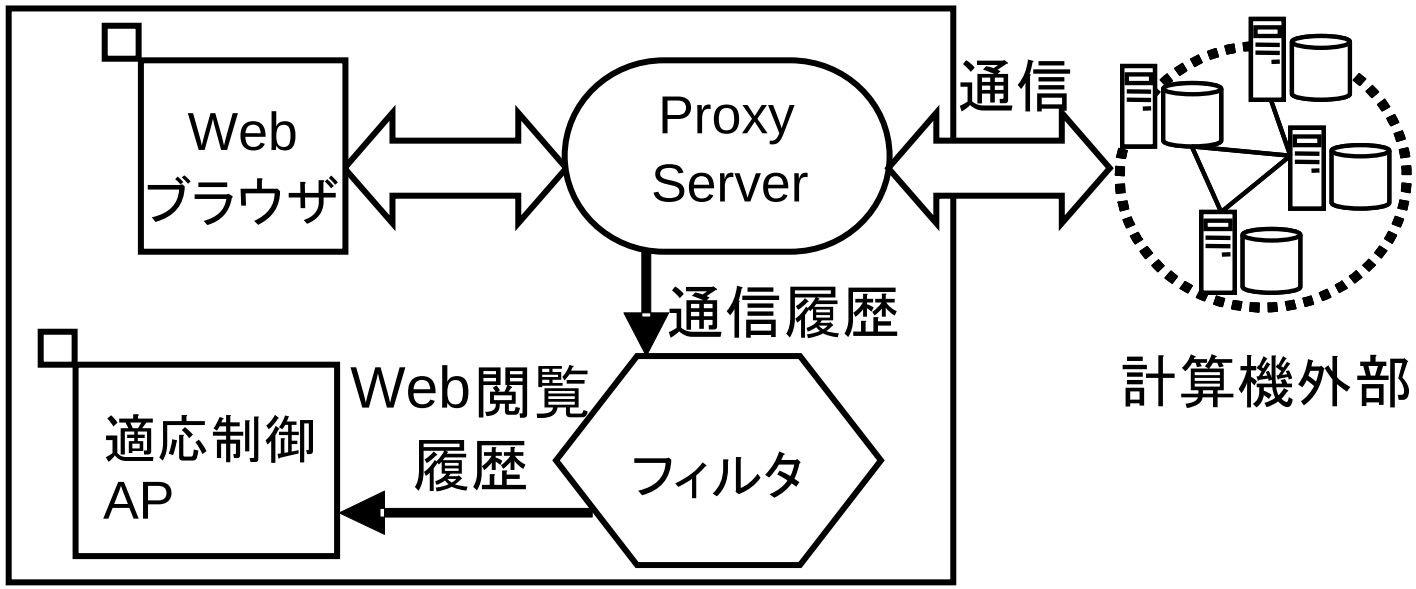
(2) データ操作機能の実装

(3) 仕事の判別方法の考案

(4) 過去の抽出データの利用方法の考案

- 参考資料

Private Proxy Server



Private Proxy Server

・利用者計算機上でプロキシサーバとなり計算機外部の間の通信を仲介

・HTTPのGETリクエストに関するすべての履歴を取得

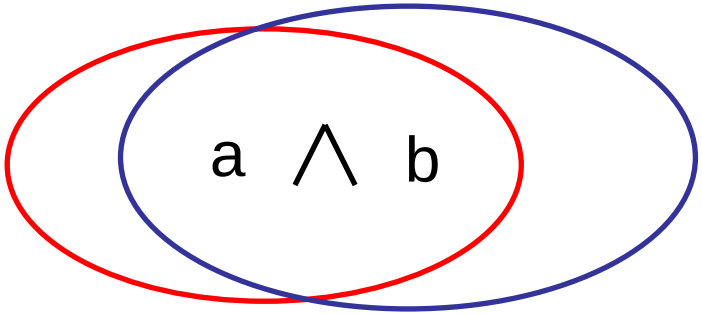
データの再現率と適合率

「パス取得漏れの少なさ」の評価尺度：データの再現率

「不要なパス取得の少なさ」の評価尺度：データの適合率

$$\text{データの再現率} = \frac{a \wedge b}{a}$$

$$\text{データの適合率} = \frac{a \wedge b}{b}$$



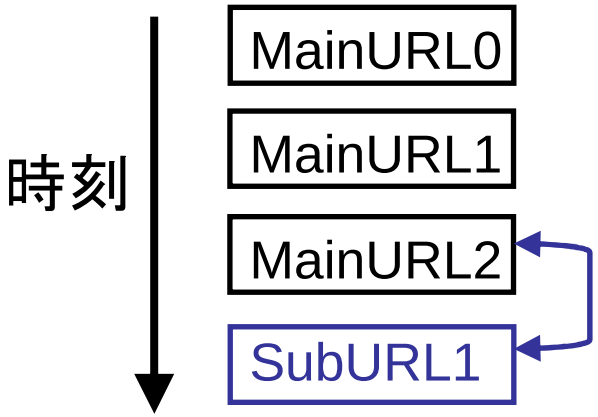
a ＝正解データ

b ＝抽出したデータ

SubURL1に関する考察

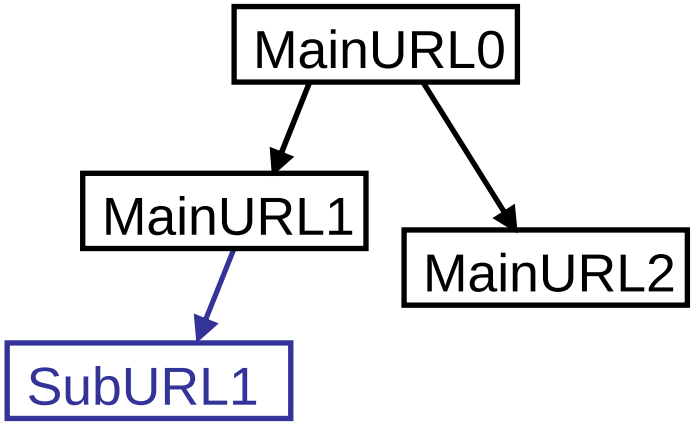
SubURL1：改良後のフィルタで抽出した誤ったデータ(E)

<改良前のフィルタ>



直前の履歴との参照時刻差が機械参照閾値より短い
SubURLと推測し，排除

<改良後のフィルタ>



機械参照閾値より長い
MainURLと推測し，閲覧時間が閲覧閾値より長いため抽出

プロキシサーバのWeb閲覧履歴

localhost - - [13/Nov/2006:11:32:38 東京 (標準時)]
（1）　　（2）

“GET http://www.it.example.com/index.html HTTP/1.1“ 200 10790
（3）　　（4）　　（5）　（6）（7）

http://www.example.com/
（8）

-> http://www.it.example.com/index.html
（4）

（1）利用者計算機のIP
（2）参照した日時
（3）リクエストメソッド
（4）リクエストURL
（5）HTTPのバージョン
（6）サービス状態コード
（7）受信したバイト
（8）リファラ