時制データベースにおける時区間結合及びそれに基づく時制データの集約手法

成　凱† 　稲永 健太郎†

†九州産業大学情報科学部 〒813-8503 福岡県福岡市東区 2-3-1
E-mail: †{chengk, inenaga}@is.kyusan-u.ac.jp

あらまし 近年，時間的制約が存在する事象を表す時制データが急増し，大規模の時制データの効率的な処理技術が求められている．本稿では，時制データベース集約問題の一つである空時区間報告問題を取り上げ，その数学基礎とアルゴリズムを考案する．時区間報告問題とは，問合せ範囲[p,q]において一定の長さ α 以上の空時区間を算出する問題である．従来手法では，時区間の端点からなる基本時区間上で集約するため，計算コストが高く集約結果のサイズが大きい問題点がある．本稿では，時区間結合，時区間短縮により，空時区間報告問題を串刺し報告問題に帰着させ，さらに串刺しグループ化によって効率よく解決することを示す．提案手法の有効性を検証するため，評価実験を行った．

キーワード 時制データベース集約，空時区間報告，串刺しグループ化，串刺し報告，時区間結合，時区間短縮

1. はじめに

時制データとは，ある事象について時間的制約が存在し，その事象の開始と終了の時間や時間の継続を保持しているようなデータである．電子カルテ，ライフログ，施設予約などの応用分野で時制データが重要であり，時制データの効率的な処理技術が必要不可欠である．Google カレンダーのようなオンラインスケジュール管理システムでは，用件名，開始時刻，終了時刻等を簡単に登録し，個人のスケジュール管理に加え，家族間や職場で共有することで，共同作業を進めやすく仕事の効率が高くなる利点がある．しかし，複数のメンバーが関わる新しい予定を立てるとき，メンバー全員の都合の良い時間帯を見つけるため，全員の予定を調べ回って結果を集計しないといけない．さらに，全員の都合のより時間帯が見つからない場合や重要人物の都合を優先させる必要のある場合がある．

時制データベースにおける時区間を軸とする集約演算は時制集約(temporal aggregation)として知られ，これまでは多くの研究が存在している．既存の研究[2]では，集約対象のグループ化に固定長区間 (fixed intervals)または不変区間(constant intervals)が用いられる．固定長区間の場合は，時間軸を一定の長さ，通常は時間単位(例えば，年度，四半期，月)にしグループ化する．集約は各グループに対して行う．一方，不変区間では，集約結果が不変である最大の時区間をグループ化の基準とする．固定長区間は区間の長さを適切に決めるのが難しい問題があり，不変区間は時区間の分割が細かすぎて集約結果が元データより大きい問題があると知られている[2][3].

本稿では，時制データベースにおいて一定の長さをもつ空き時間を集約する空時区間報告問題を中心に，時制データの効率的集約手法を提案する．

2. 問題定義

時制データベースとは時間情報をもつ一連の事象を管理するデータベースとし，各事象に時区間 (time interval) がついており，その事象の存在期間や有効期間を示している．本節では，本研究の問題及びそれに関連する基本事項を説明する．

2.1 時区間

時区間とは時間定義域上の連続な区間である．時区間 $t = [t_s, t_f]$ は開始時刻 t_s と終了時刻 t_f をもつ時間定義域上の閉区間を表す．時区間を線分で表し，線分の端点はそれぞれ開始時刻，終了時刻を示し，線分の長さ，位置関係から時区間の相対関係がわかる．

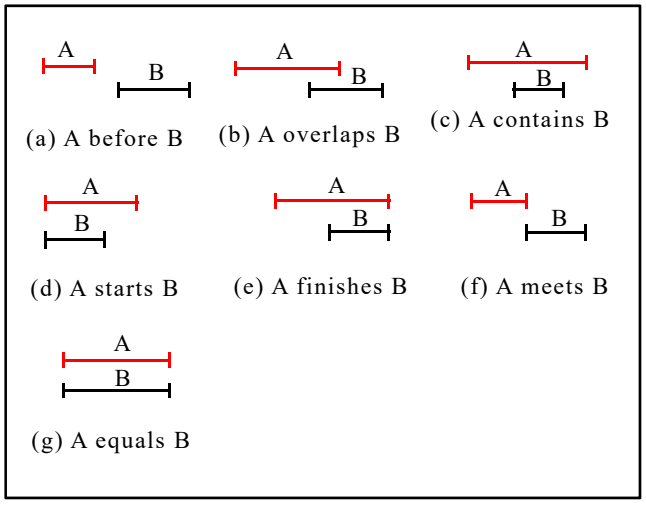

図 1 時区間の相対関係

Allen[1]は二つの時区間の間に 13 種の時間的関係が定義できることを示した．図 1 は時区間 A と B の間に存在する時間的関係を示している．(a)～(f)は片方のみを示しているが，逆関係も成り立つため全部で 13 種類の関係がある．例えば，図 1(a)では A before B，つまり A が B より前である関係を示しているが，逆に B after A，B が A より後であるという関係も存在する．

$E = \{e_1, e_2, \cdots, e_n\}$ を事象識別子の集合とするとき，$(e_i, s_i, f_i) \in E \times N \times N$を事象時区間(単に「事象時間」)という．N が時間定義域であり，s_iと f_iはそれぞれ事象の開始時刻と終了時刻である．事象時区間は，事象の存在を表す「実時区間(real interval)」，事象の休止を表す「空時区間(null interval)」に区別する[9].

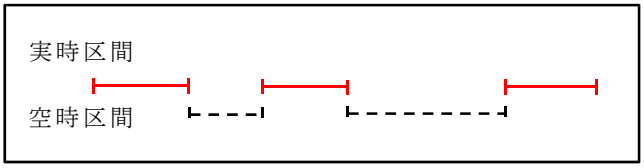

図 2 実時区間・空時区間

本論文では，特に説明しない限り，事象識別子を意識しなくて事象時区間と時区間は区別せずに使う．また，同一事象の時区間が互いに交わりを持たないとし，複数の事象からなる時区間集合には，交わりが存在するとする．

2.2 時制集約問題

時区間集合 I に対し，以下の時制集約問題を考える．

（１）空時区間報告問題(null time reporting)

問合せ範囲[p,q]，パラメータ α が与えられたとき，I に含まれる長さ α 以上の空時区間を全て報告する．

例 1 ：1 時間半の打合せを行いたいとき，メンバー10 人全員の予定の入っていない最低 1 時間半の空時区間を見つけるのが，空時区間報告問題の例である．

時区間報告問題は以下の区間串刺し報告(interval stabbing reporting)に帰着できる．問合せ範囲[p,q]において，$[p,q] \cap [s,f] \neq \emptyset$ を満たすような I の要素[s,f]を全て求めて報告する．長さ α という状況をうまく吸収することで，α を考慮せずに問題を解決できる．

区間串刺し報告問題は以下の**点串刺し報告問題**(stabbing reporting)の拡張である．問合せ時間点 $q \in N$ に対し，$q \in [s,f]$ を満たすような I の要素[s,f] を全て求めて報告する．

（２）空時区間計数問題(null time counting)

問合せ範囲[p,q]，パラメータ α が与えられたとき，I に含まれる長さ α 以上の時区間の数を報告する．

例 2 ：1 時間半の打合せを行いたいが，メンバー10 人全員参加のうち，8 名以上参加できる 1 時間半以上の空時区間を見つける問題が空時区間計数問題の例である．

空時区間計数問題は以下のような区間串刺し計数(interval stabbing counting problem)に変換できる．問合せ区間[p,q]に対し，$[p,q] \cap [s,f] \neq \emptyset$ を満たすような I の要素[s,f]の数を求めて報告する．

区間串刺し計数問題は点串刺し計数問題(stabbing counting)の拡張である：問合せ時間点 $q \in N$ に対し，$q \in [s,f]$ を満たすような I の要素[s,f]の数を求めて報告する．

3. 時区間集合上の演算

3.1 時区間結合

接続している時区間が結合可能(coalescible)な時区間といい，連続発生する事象を意味する．図 3(a)（左側）に示すように，時区間 A と B が接続している（A meets B）から，A，B 結合後一つの時区間になる．時区間が厳密に接続していないが，かなり近い（距離 α 以内）ならば，結合可能とみなすとき，α **結合可能**(α coalescible)という．図 3(b)（右側）は α 結合可能の例である．

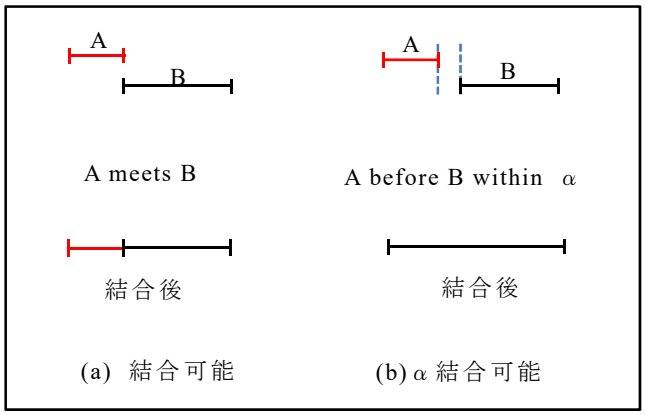

図 3 時区間結合

時区間結合(temporal coalescing)は，時制データベースに断片化された時区間を処理し，時制データベースの最適化技術として用いられている[7]．本稿では，これまでと違い，α 結合可能を導入し，不必要な処理コストを削減し，時制集約の効率化を目的としている．

時区間集計のために時区間集合上の演算を説明する．演算の詳細を述べる前に，時区間の間の時間関係を理解する必要がある．

3.2 時区間短縮

時区間結合と合わせて存在するのが，時区間短縮である．時区間短縮とは時区間の両端より一定の長さを縮めることである．時区間の α 結合可能に対応して，時区間の長さ α 以上であるとき，α 短縮可能と定義する．時区間[s,f]の α 短縮は以下の式で行う．

短縮前：$[s, f]$

短縮後：$[s+(1-\lambda)\alpha, f-\lambda\alpha],\ (0 \le \lambda \le 1)$

図 4 はα短縮のイメージを示している．元の時区間が矢印の方向に左端より$(1-\lambda)\alpha$，右端より$\lambda\alpha$を短縮しており，短縮された部分は$(1-\lambda)\alpha+\lambda\alpha=\alpha$である．

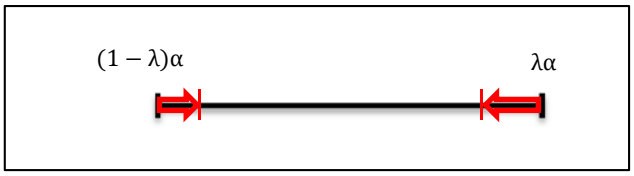

図 4 時区間短縮（両端より長さα分だけ縮める）

時区間短縮により，長さ未満の空時区間をあらかじめ集約対象から排除し，必要のない計算コストを省くことができるだけではなく，後で述べるように，空き時間集約の簡素化にもつながる．

3.3 時区間の集合演算

時区間を要素とする集合に対して，二つの時区間の和集合と差集合を求める演算（和演算 A+B，差演算 A-B）は表 1 に示している．時区間の時間関係により，結果が違う．(a)，(f)には A と B の重なりがないが，(b)～(e)では A と B の一部が重なっている．(c)では，時区間 A が B により分断され，二つの区間になっている．和演算は事象データベースから空きではない時区間を求めるために利用される．差演算は，実時区間より空時区間を求めるに使われる．

表 1　$\boldsymbol{A}=[a_s, a_f]$と$\boldsymbol{B}=[b_s, b_f]$の集合演算

時制関係	A + B	A − B
A before B	A, B	A
A overlaps B	$[a_s, b_f]$	$[a_s, b_s]$
A contains B	A	$[a_s, b_s], [b_f, a_f]$
A starts B	B	∅
A finishes B	B	∅
A meets B	$[a_s, b_f]$	$[a_s, a_f]$
A equals B	A	∅

4. 空き時間の評価

問合せ区間$Q=[p, q]$に長さα以上の空時区間を求めるために，問合せ時区間と交差するすべての事象の空時区間をデータベースより取り出し，結果を集約する．データベースに実時区間と空時区間両方を検索できると仮定する．実時区間しか保存されていない場合，図 2 に示すように空時区間を求められる．

4.1 問合せ区間における空時区間抽出

空時区間 X=$[s, f]$が問合せ範囲$Q=[p, q]$との共通部分で長さα以上の時区間は，以下のいずれかの条件を満たす必要がある．

1）．X の前半部分長さα以上が Q の範囲内にある

$$p-\alpha \le s \le q-\alpha$$

2）．X の後半部分長さα以上が Q の範囲内に入っている．

$$p+\alpha \le f \le q+\alpha$$

上記の条件は以下のように書き換えられる．

$$[s+(1-\lambda)\alpha, f-\lambda\alpha] \cap [p, q] \neq \emptyset, \qquad (1)$$

λは$[0,1]$以内の任意の実数である．例えば，

$\lambda=1$のとき，$[s, f-\alpha] \cap [p, q] \neq \emptyset$, (2)

$\lambda=0$のとき，$[s+\alpha, f] \cap [p, q] \neq \emptyset$, (3)

したがって，対象の空時区間を全て同じ方法で長さαだけ縮めると，空時区間報告問題が串刺し報告問題に帰着できる．また，事象の実時区間から見ると，これは実時区間を長さαだけ伸ばす結果であり，延長により，実時区間が接続できるようになる．このとき．元の実時区間がα結合可能でなければならない．

データベースから上記の条件でαだけ縮まった空時区間を検索し，開始時刻の昇順，終了時刻の降順で並べ替えておく．

4.2 空時区間の時制集約

（1）時区間グループ化

空時区間抽出で，問合せ区間内の事象のαだけ縮まった空時区間を整列済みの形で受け取りながら，次のように区間をグループ化していく．x_{left}とx_{right}をそれぞれ今まで処理した時区間の開始時刻の最大値と，終了時刻の最初値とする．処理が進むにつれ，x_{left}とx_{right}が段々近づいていき，$x_{left} < x_{right}$になる直前までの時区間をグループI_{mid}にし，区間$[x_{left}, x_{right}]$をグループI_{mid}の代表とする．

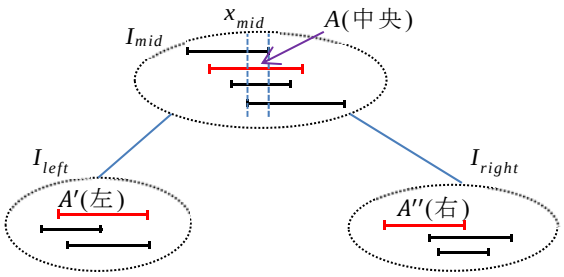

図 5 時区間の串刺しグループ化

さらに，図 5 に示すように，終了時刻がx_{left}より小さな時区間集合をI_{left}，開始時刻がx_{right}より大きな時区間集合をI_{right}とし，それぞれに対して，上記と同じ手順でグループ化する．

さらに，グループ代表$[x_{left}, x_{right}]$が時間軸上で完全に分離していることから，二分探索木のようなデータ構造にグループ代表を格納することができる．これは点串刺し報告問題に適した区間木(interval tree)に似ているので，串刺しグループ化(stabbing grouping)と呼ぶ．

区間を頂点とし区間同士の重なりを辺とするようなグラフは区間グラフ(interval graph)といい，インターバルグラフからクリーク(clique)を抽出する問題になる．図 6 は 5 つの区間 A，B，C，D，E と，その重なり関係を表す区間グラフを示している．そのうち，{B, C，D}，{C，D，E}，{A，B}がクリークである．

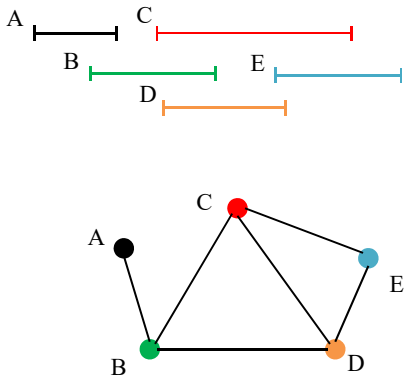

図 6 区間の重なり関係を表す区間グラフ

区間グラフでは同一時区間が複数のクリークに参加できることから，時区間の串刺しグループ化では，一つの時区間が複数のグループに参加できる．図 5 では区間 A がグループI_{mid}に入っており，グループ代表と重なる部分を除いて，残りのA'とA''がさらに左右の子ノードに参加できる．

（2）グループ別集約

グループ代表の時区間$[x_{left}, x_{right}]$はグループの共通空時区間であり，α だけ縮まったことから，実際の共通空時区間が α 以上であることが保証される．この時区間集合を Ω とし，グループ・サイズの大きい順に並べ替えておく．各問題の解決に活用する．

空時区間計数：問合せ区間[p, q]，パラメータ α が与えられたとき，串刺しグループと関連する事象数を報告するとよい．

空時区間報告：問合せ区間[p, q]，パラメータ α が与えられたとき，串刺しグループと関連する事象の事象識別子を出力すればよい．

空き時間を出力する前に，長さ，曜日，時間帯（午前か午後）などの要素を考慮して優先度を総合的に評価し，優先度の高い順で出力される．

また，事象によって空時区間に重みを付ける必要があるとき，グループ・サイズのかわりに，重みの合計を使うと良い．

定理 1：短縮前の時区間集合に長さ α 以上の共通部分を持つ必要十分な条件は，短縮後の時区間集合の共通部分が空ではないことである

証明：短縮前の時区間集合に長さ α 以上の共通部分を持つとき，短縮によりその共通部分から減らされた部分が α 以下なので，結果として，短縮後の時区間集合の共通部分が空ではない．

逆に，短縮後の時区間集合の共通部分が空ではないとき，短縮前の時区間は共通部分を中心に最低長さ α がプラスするので，元の時区間集合が長さ α 以上の共通部分を持つ．

4.3 串刺し報告アルゴリズム（BTA）

本節では，串刺し報告アルゴリズム BTA（Stabbing Temporal Aggregation）を設計する．データ構造に AVL 木を使い，各ノードが一つのグループと対応させ，グループ代表及びグループ関連の情報をノードに格納される．図 7 はアルゴリズムの具体的手順を示している．

α 短縮の空時区間I_iを次のように処理し，串刺しグループ化する．まず，AVL 木の根ノードから，現在ノードのグループ代表ωとI_iの共通部分を求め，その結果により処理が分岐する．（1）I_i before ωなら，現在ノードの左の子に対して探索を続け，左へ行き詰ったら，I_i をもつノードを新規追加する．（2）I_i after ωの場合，現在ノードの右の子に対して同じ探索を続け，右へ行き詰ったら，I_i をもつノードを新規追加する．（3）I_i と ωが共通部分を持っている場合，I_i の事象識別子を現在ノードに登録し，必要に応じてグループ代表を更新する．表 1 に従い$I_i - \omega$の結果を求め，現在ノードの左の子，または，右ノ子にさらに探索しながらノードの追加・更新を行う．

串刺し計数の場合は，AVL 木のノードにグループ代表とカウントだけ格納し，串刺し報告に必要な事象識別子は保存しなくて良い．

n 個の時区間を処理するコストを考察する．AVL 木におけるノードの挿入・探索に$O(\log n)$の計算量が必要であり，区間の分裂の数は定数とみなして，全体の時間計算量は$O(n \log n)$である．また，串刺しグループを格納するための AVL 木の空間は平均$O(n)$である．

串刺し報告アルゴリズム（BTA）

１）.（初期化）i=0; j=0;

AVL 木 $= \emptyset$, $\omega = [s', f'] = [0, +\infty]$

２）.（走査）i = i + 1; $I_i = [s_i, f_i]$

３）.（ωとI_iの共通部分を求める）

$s = MAX(s', s_i)$, $f = MIN(f', f_i)$

４）.もし$s \leq f$ ならば（共通部分存在）

$\omega = [s, f]$とI_iの事象識別子を合わせてノードとして AVL 木に追加する．さらに$I_i - \omega$を求め，ステップ３）に戻り，$I_i - \omega$を AVL 木に追加を続ける

５）.もし$s > f$ であれば（共通部分なし）

・I_i before ω: 左の子に対して探索を続け，左へ行き詰ったら，I_i を新しいノードとして追加

・I_i after ω: 右の子に対して探索を続け，右へ行き詰ったら，I_i を新しいノードとして追加

・ステップ２）へ戻る

６）.（出力）

AVL 木を中間順で走査し,各ノードに格納されているグループ代表及び事象識別子集合を出力する．

図 7 時区間集約アルゴリズム（BTA）

5. 関連研究

本研究に関連して，時制データベース集約の研究が多く存在している[2][3][5]．これまでの研究は大きく分けて，時間点ベースの時制集約ITA（instant temporal aggregation）と時間帯ベースの時制集約STA（span temporal aggregation）がある．STAは固定長区間(fixed intervals)を基本とし，時間軸を一定の長さに分割しグループ化する．本研究の空時区間報告問題に適していない．一方，ITAは集約対象のグループ化に不変区間(constant intervals)が用いられ，集約結果が不変である最大の時区間をグループ化の基準とする．不変区間は時区間の分割が細かすぎて集約結果が元データより大きいという問題が知られている．PTA[3]は上記の中間に位置するが，STAと同様，本研究の問題に適していない．

問合せ結果の空判定問題(emptiness query)[6]はデータベースにける問合せ最適化の技術として開発されている．結果が空になる問合せを早いうちに判断し，無駄な処理を省く技術である．本研究の目的は，空時区間を求めるためであり，空判定問題の技術は適用できない．

また，先行研究[8]では我々は時区間結合の概念に基づく近似的時制集約の提案をしたが，集約アルゴリズムが単純なものであった．本研究はその発展として串刺しグループ化による集約を提案した．空時区間の概念は[9]を参考にした．

6. 実験評価

本章では，提案手法の有効性を検証するため，評価実験を行う．

本研究では，空時区間から一定の長さ以上の共通空き時間を求める時制集約問題を串刺し報告問題に帰着させ，串刺しグループ化によってより効率的に空き時間を実現できる．提案手法を評価するために，評価実験を行った．評価対象は以下である．

(1) Stabbing Temporal Aggregation (BTA)

図 7 の串刺しグループ化と AVL 木に基づくアルゴリズムを実現したもので AVL 木の平衡係数が 1 である．ノードにグループ代表となる時区間とともに，事象カウンタ(串刺し計数問題), または, 事象識別子リスト(串刺し報告問題)を格納している．

(2) Instant Temporal Aggregation (ITA)

対象となるすべての時区間の端点をベースにした基本時区間(elementary intervals)に対して集計を行い，出力する前に,隣接して結合可能な区間を結合させる．

実験はすべて物理メモリ 8GB，64 ビット Windows10 環境下で行った．実験では空き時間検索の効率性について評価を行った．問合せ範囲[p,q]を選択率(query selectivity)として変動し，それぞれの選択率に対して，実行時間及びメモリ使用量を計測した．実行時間の内訳を調べるために, DB 検索時間, 集約処理時間をそれぞれ計測した．各選択率に対し同じ処理 10 回繰り返した時間やメモリ使用量の平均を最終結果としている．

6.1 実験データ

提案手法を評価するために，以下のように実験用事象データベースを生成した．時間領域を[1..10^6], 事象識別子を[1..1,000]からの整数として，各事象識別子に対し，時区間を 1,000 個発生し，各データセットに 1,000,000 行のデータが含まれている．各時区間の長さや次の区間との間隔の長さとも乱数として発生する．

例：[298,304],[532,1006],[1174,1482],…

実時区間の発生と同時に，実時区間の間の区間を空時区間として求め，空時区間テーブル(null_time)に追加しておく．

例：[0, 298], [304,532], [1006,1174],…

長さや間隔ともに均一になるようなデータセット Uniform と指数分布に従うデータセット Exponential を２セット作成した．指数分布のデータセットは中間値

が 2,000 であり，短い時区間が多く含まれている．図 8 はこれらのデータセットのヒストグラムを示している．

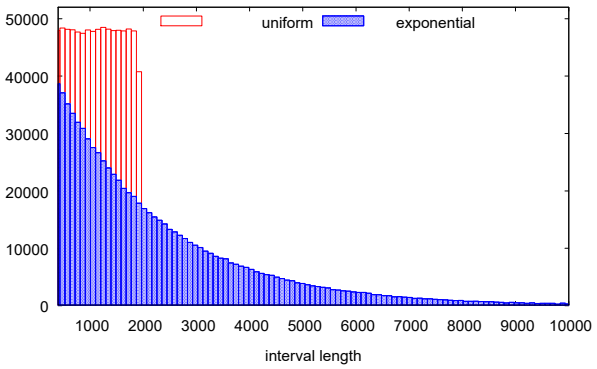

図 8　実験用データセット

DBMS は MySQL 5.5 を使用しており，MySQL では SQL キャッシュがデフォルトに機能しているため，同じ問合せが繰り返し実行するときキャッシュ結果が利用される．　問合せ SQL は SQL_NO_CACHE によりキャッシュ禁止としている．

事象データベーススキーマ(eid, s, f)である．

eid INT　事象識別子　1..1,000 の整数

s BIGINT 開始時刻　1..10^6 の整数

f BIGINT 終了時刻　1..10^6 の整数

6.2 実験結果

主な結果は以下の通りである．

1. **総実行時間**，つまり，データベース検索と集約処理を合わせた実行時間である．図 9～図 12 に示すように，選択率が大きくなるにつれ，処理時間も上昇傾向であるが，BTA は串刺し報告問題と計数問題ともに，ITA より優れた結果を確認できた．
2. DB 検索時間，つかり，データベースから問合せ範囲に重なる時区間を抽出する時間は，大きな違いが見られなかった（図 13～図 16）．
3. 集約処理時間，つまり，DB 検索結果となる対象時区間を集約し，集約結果を報告する時間である．図 17～図 20 より集約処理時間は BTA が ITA より優れていることが分かった．
4. メモリ使用量．実行時のメモリ使用量を計測した．結果は図 21～図 24 に示されている．BTA は，処理時間の短縮に効果があるだけではなく，メモリ使用量も少ないことがわかった．

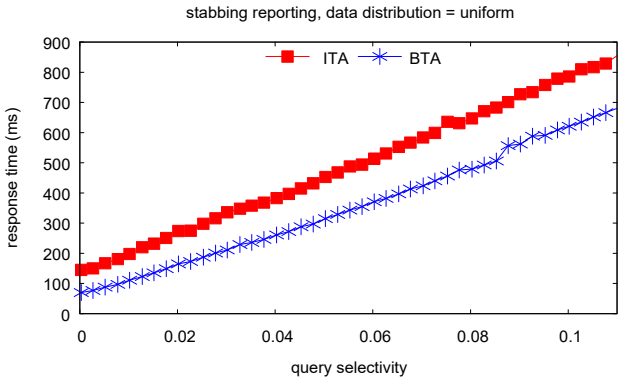

図 9　串刺し報告問題の総実行時間(Uniform)

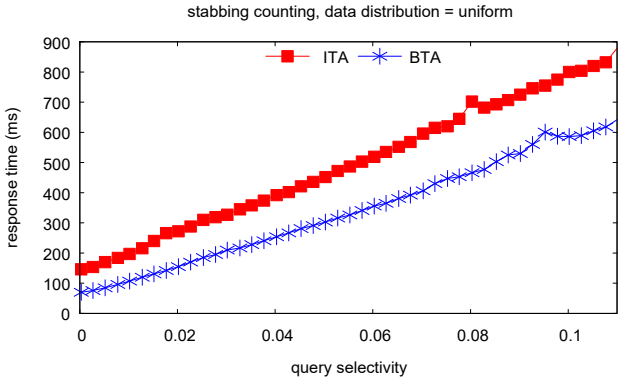

図 10　串刺し計数問題の総実行時間(Uniform)

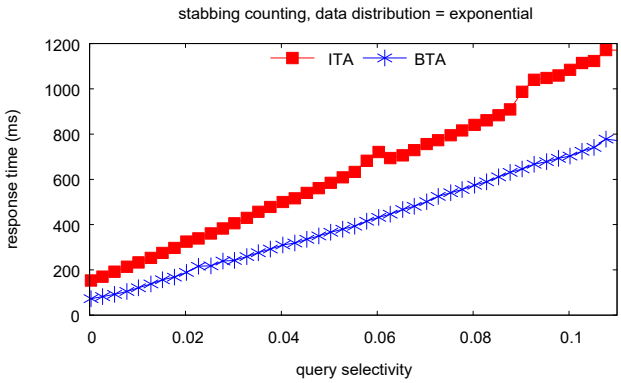

図 11 串刺し報告問題の総実行時間(Exponential)

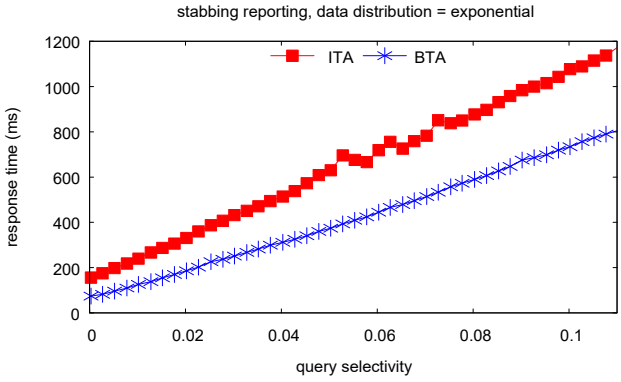

図 12 串刺し係数問題の総実行時間(Exponential)

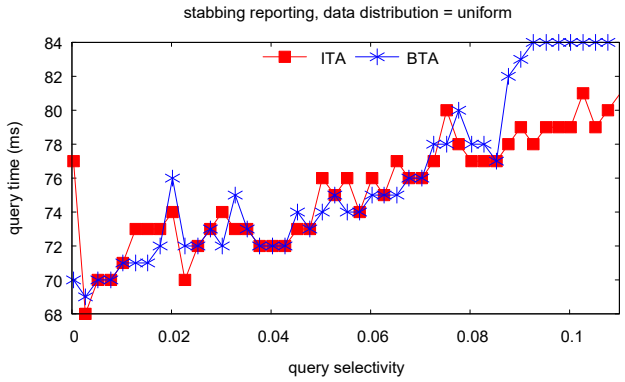

図 13　串刺し計数問題の DB 検索時間(Uniform)

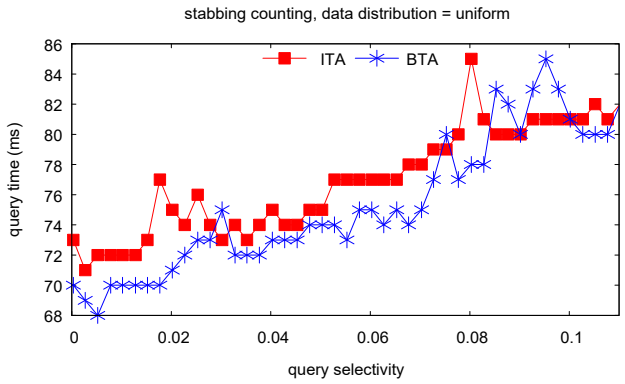

図 14　串刺し報告問題の DB 検索時間(Uniform)

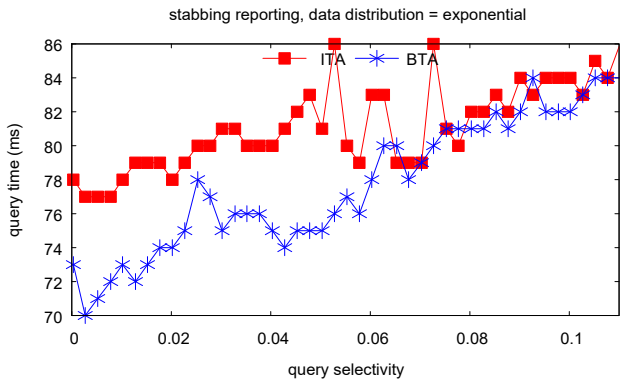

図 15 串刺し報告問題の DB 検索時間(Exponential)

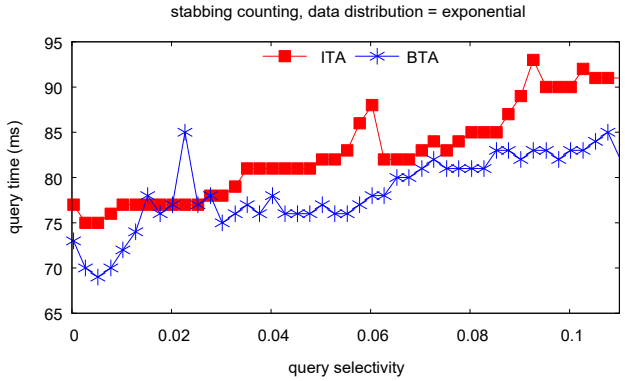

図 16 串刺し計数問題の DB 検索時間(Exponential)

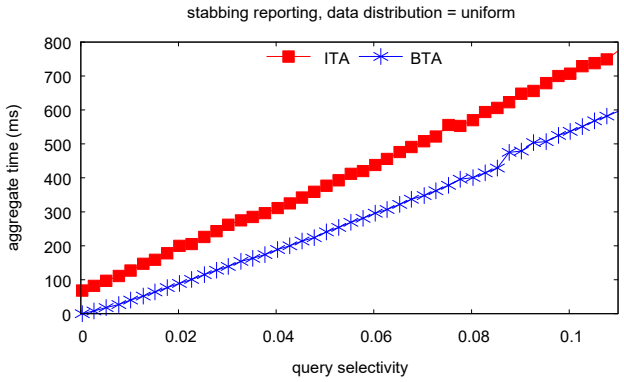

図 17　串刺し報告問題の集約時間(Uniform)

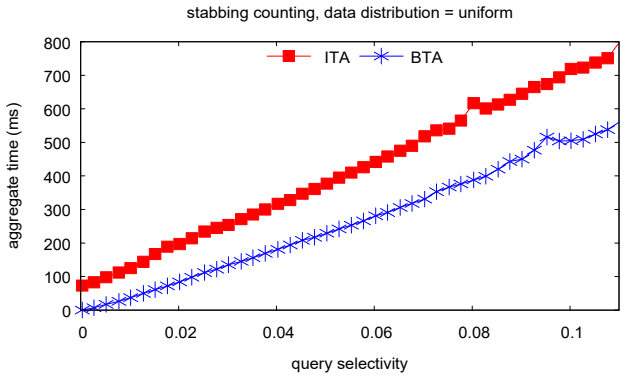

図 18　串刺し計数問題の集約時間(Uniform)

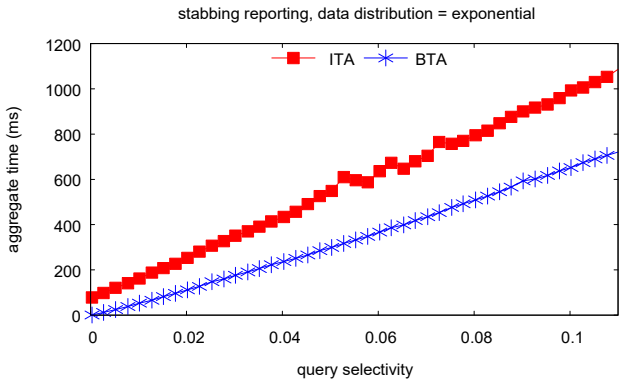

図 19　串刺し報告問題の集約時間(Exponential)

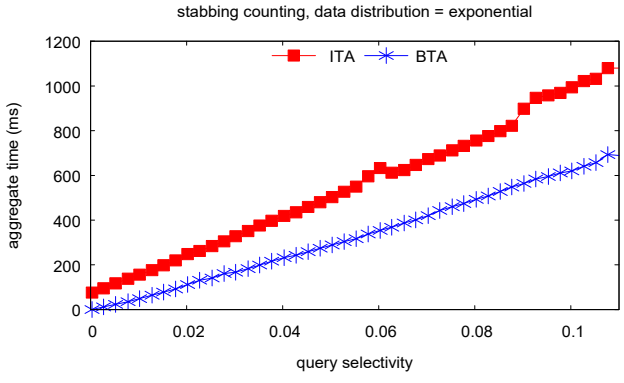

図 20　串刺し計数問題の集約時間(Exponential)

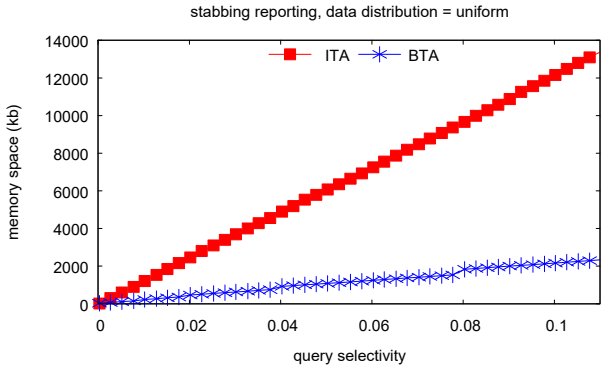

図 21　串刺し報告問題のメモリ使用量(Uniform)

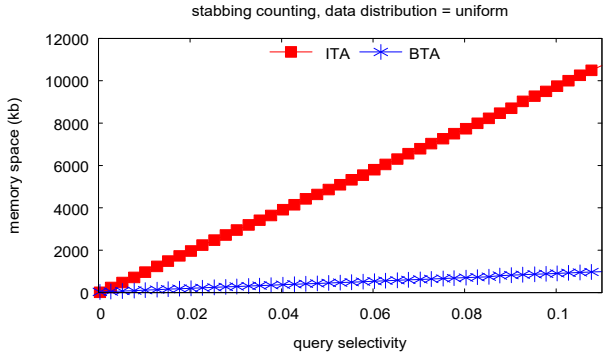

図 22　串刺し計数問題のメモリ使用量(Uniform)

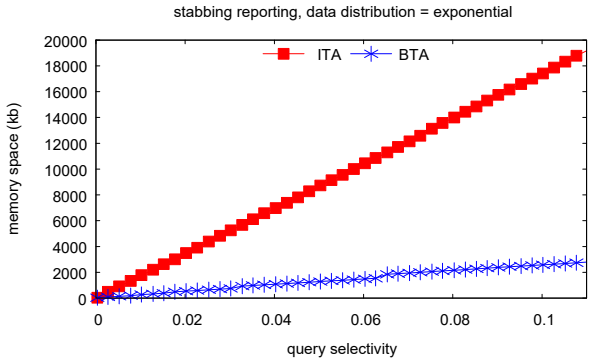

図 23 串刺し報告問題のメモリ使用量 (Exponential)

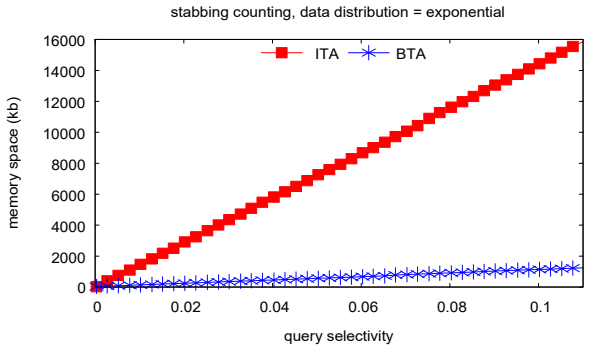

図 24 串刺し計数問題のメモリ使用量 (Exponential)

以上の結果から，提案手法の有用性を検証できた．評価対象の ITA は時区間を細かく分割し，それぞれに対して集計をしないといけないので，長い処理時間と大きなメモリ使用量を消費してしまうことが考えられる．

7. おわりに

本研究では時制データベースから一定の長さ以上の共通空き時間を求める時制集約問題に着目し，時区間結合，時区間短縮により，この問題を空時区間上の串刺し報告問題に帰着させ，データベースシステムの既存の機能を活かしより効率的な空き時間検索ができた．

今後の課題として，空時区間に適した索引機構の開発が考えられる．また，グループ化の手法はパラメータ α に依存するものであり，再計算する必要がある．パラメータに依存しない手法を検討していく．

謝辞

本研究は九州産業大学産業経営研究所よりご支援を頂いた．また，研究を進めるにあたり，朝廣雄一教授に様々な助言をいただいた．

参考文献

[1] J. Allen. Maintaining Knowledge about Temporal Intervals. Communications of ACM, vol.26, issue 11, pp.832-843, 1983.

[2] Lopez, Ines Fernando Vega, Richard T. Snodgrass, and Bongki Moon. Spatiotemporal aggregate computation: A survey. IEEE Transactions on Knowledge and Data Engineering, Vol.17 No.2 (2005): 271-286.

[3] Juozas Gordevičius, Johann Gamper, and Michael Böhlen. 2012. Parsimonious temporal aggregation. The VLDB Journal 21, 3 (June 2012), pp. 309-332.

[4] Chen, Yi-Cheng, et al. An efficient algorithm for mining time interval-based patterns in large database. Proceedings of the 19th ACM international conference on Information and knowledge management (CIKM 2010). pp. 49-58, 2010.

[5] Yang, J., and Widom, J. Incremental computation and maintenance of temporal aggregates. In Proceedings of the 17th International Conference on Data Engineering (ICDE 2001). (pp. 51-60). IEEE.2001.

[6] Mayank Goswami, Allan Grønlund, Kasper Green Larsen, and Rasmus Pagh. Approximate range emptiness in constant time and optimal space. (SODA '15). Pp.769-775.

[7] Zhou, Xin, Fusheng Wang, and Carlo Zaniolo. Efficient temporal coalescing query support in relational database systems. *International Conference on Database and Expert Systems Applications*. Springer Berlin Heidelberg, 2006

[8] Kai Cheng. Approximate Temporal Aggregation with Nearby Coalescing, DEXA (2) 2016: 426-433.

[9] 天笠 俊之，田頭 利規，金森 吉成，増永 良文．制約を導入した時区間代数．電子情報通信学会技術研究報告．DE，データ工学 95(148), 41-48, 1995-07-19