分析問合せ処理の資源律速ならびに消費電力の特性に関する考察

羅　博明† 早水　悠登†† 合田　和生†† 喜連川　優††,†††

† 東京大学大学院情報理工学系研究科 〒 113–8656 東京都文京区本郷 7–3–1
†† 東京大学生産技術研究所 〒 153–0041 東京都目黒区駒場 4–6–1
††† 国立情報学研究所 〒 100–1003 東京都千代田区一ツ橋 2–1–2
E-mail: †{luo,haya,kgoda,kitsure}@tkl.iis.u-tokyo.ac.jp

あらまし データセンタにおける消費電力量拡大を受け，省電力性を考慮したデータ処理が注目を集めている．分析問合せは現在もデータドリブンな意志決定等に利用され続けており，そのワークロードとしての傾向は資源律速の違いにより変化することが知られているため，問合せ最適化で省電力性を考慮するためには資源律速の違いに合わせた適切なモデルを作成することが必要である．しかし，資源律速が切替る中間領域に関する特性，特にサーバの消費電力への影響は未だに詳しい調査が行われていない．本論文では，中間領域を含めた資源律速の変化に伴う特性の解明を目的とする．読み出し中心である分析問合せを多重的に処理するワークロードにおいて，問合せが対象とする読み出し範囲の広さが変化することで資源律速がプロセッサとストレージの間で入れ替わり，資源の利用率・スループット・消費電力等に大きな影響を与えることを示す．そしてその知見を利用した実行時システム制御について考察する．

キーワード データベースシステム，プロセッサ，問合せ処理，省電力化，エネルギー効率

1 はじめに

近年，データセンタにおける消費電力量増大を受けて，消費電力削減のための様々な工夫が注目を集めている．消費電力量増大の要因としては，データを蓄積・分析することの重要性が唱えられてきたなどにより，各所でデータセンタへの IT 資源の投入が行われ続けているという背景が挙げられる．天然資源防護協議会の報告では，米国におけるデータセンタの消費電力量は，2013 年には 910 億 kWh であったが，予測では 2020 年には 1400 億 kWh にまで増加するとされている [1]．同様な傾向は欧州におけるデータセンタの消費電力量にも見られ，2007 年では 560 億 kWh だったものが，2020 年には 1040 億 kWh に達すると見込まれている [2]．

データセンタにおける省電力化の工夫として，データベースシステムにおける問合せ最適化における省電力技術の適用が挙げられる．データセンタにおける主要なワークロードである分析問合せ処理を対象として，省電力性を考慮した問合せ計画を出力する試みが過去に盛んになされた [3–7]．こうした省電力性を考慮した問合せ最適化においては，ワークロードにおける性能とエネルギー消費の特性を正確に把握するために定量的なモデルの構築が必要である．著者らが過去に行ったプロセッサ動作モードが分析問合せ処理のスループット・消費電力に与える影響に関して研究 [8, 9] では，資源律速の違いによってその特性に差異が見られることを示した．ここでいう資源律速とは，プロセッサの処理速度やストレージの IO 速度などの特定の資源が律速要因となることで処理が律速されている状態を指す．分析問合せ処理では CPU 律速と IO 律速が二つの大きな資源律速の状況として存在することを明らかにし，それぞれに対して適切なモデルの構築が重要であることを確認した．しかし，CPU 律速と IO 律速の中間領域に関してはまだ調査が十分に行われていない．

本論文では，資源律速の変化に伴うスループットや消費電力に対する影響を把握することを目的とする．すなわち，中間領域を含む CPU 律速と IO 律速に跨がる特性の解明を目指す．実験により，問合せ計画がほぼ同等となるような問合せを複数個用意し多重的に処理するような環境下で，それらの問合せが走査するデータベースの範囲の広さを変化させた場合に資源律速の切替りを観測できることを示す．

本論文は次のような構成からなる．2 節では分析問合せ処理の基本的な特性に関して述べる．3 節では今回主眼を置く資源律速の変化によるワークロード特性の変化を把握するために行った実験に関して述べる．4 節では計測実験の結果とそれに対する考察を述べる．5 節では本論文の総括と今後の展望を述べる．

2 分析問合せ処理の基本的な特性

2.1 分析問合せ

分析問合せ（Analytical Query）とは，データベース上のデータに対し合算・平均等の操作を組み合わせることで複雑なデータ分析を行うような問合せのことである．データ分析を目的としているため，分析問合せではデータベースへの書き込みは基本的には行われず，読み込み主体の処理となることが知られている．応用先として意思決定システムがあり，主にビジネスの場で現在も広く使われている．代表的な分析問合せのベンチマークとしては TPC-H [10] が業界標準で用いられており，生成された顧客情報や商品情報，注文情報等を分析する問合せ

を提供する．この他にも TPC-H の代替となる TPC-DS [11] や TPC-H の派生である SSB [12] 等が存在している．

本論文では分析問合せ処理を行うシステムとして，HDD や SSD のようなストレージデバイス上にデータベースが存在するものを想定する．これは，分析問合せが対象とする問題の性質上，データベースのサイズはメインメモリに収まりきらない範囲の規模（数百 GB 以上）になることが多いためである．

2.2 プロセッサ動作モード制御が分析問合せ処理に与える影響

分析問合せではストレージからの読み込み主体のワークロードでなおかつ複雑な分析計算が行われると既に述べた．一般的にメモリに比べてストレージへのアクセスは低スループット・高レイテンシであり，問合せ処理はストレージ IO で律速するか，あるいはプロセッサ上での演算で律速すると考えることができる．すなわち，分析問合せ処理のワークロードは，「IO 律速」となるか，「CPU 律速」となるかに二分可能である．

著者らはプロセッサ動作モード制御が分析問合せ処理のスループット・消費電力に与える影響について，用いる問合せによってその傾向が異なることを示したが，その主な要因として資源律速の違いを挙げた． [8]．プロセッサ動作モード制御は，Dynamic Voltage and Frequency Scaling（DVFS）と呼ばれる機能によって CPU コアの動作周波数を変更することにより CPU の処理速度と消費電力を調整することを指す．傾向の違いとして，CPU 律速の問合せ処理では動作周波数が高いほどエネルギー効率が良く，IO 律速の問合せ処理では動作周波数を下げるほどエネルギー効率が良くなることを確認した．更にこの結果を踏まえて，全表走査を行う問合せを対象に動作周波数に対する消費エネルギーのモデル化が可能であることを示した [9]．

2.3 分析問合せ処理と資源律速

著者らの分析問合せ処理を対象としたプロセッサ動作モード制御に関する研究は，CPU 動作周波数を変化させた時の処理への影響を把握し，適切なプロセッサ動作モード制御を行うことを目的としていた．そしてその過程で，適切な制御のためには資源律速について把握することが重要であることを示した．

本論文ではその重要性が確認された資源律速そのものに着目する．先述した著者らの研究では，問合せをデータベースシステムに与えた時に生成される問合せ計画が異なるような問合せ同士を比較した際に，資源律速に違いを確認した．

一方で，問合せプランはほぼ同じで対象とするデータベースの走査範囲が異なる（SQL における `where` 節が異なる）ような問合せを対象として議論を行いたい．もしシングルプロセスでそれぞれ処理した場合には，読み出しサイズや読み出しの特性が同等であることが予想され，ワークロードとして同様な傾向を示すはずである．次に多重度付で動作した時，つまり複数の問合せが同時に処理されている状況を考える．この時，発行される問合せのセットとして `where` 節のみが異なる問合せを寄せ集めたものを利用した時のことを考えると，走査対象となるデータベースの範囲は `where` 節が異なる問合せの種類数により変化するはずである．つまり，ここでいう種類数が比較的小さい時にはデータベースのうちの狭い範囲を走査することになり，逆に種類数が多い時には前者と比べてデータベースのうちの広い範囲を走査することになる．

前者と後者では，必要となるプロセッサの仕事量はほとんど変わらないが，必要となる IO サイズに違いがあるため，ワークロードとして異なる動作傾向を示すことが予想できる．よって，このような設定下では問合せの種類数を変化させることにより，CPU 律速と IO 律速の切替りを観測することができると考えられ，更に中間領域に関してもその傾向を把握することが可能であると期待される．

表 1: システムのハードウェア構成．

CPU	Intel Xeon Gold 6140 @ 2.30 GHz × 2
メモリ	32GB (DDR4 8192 MB 2666 MHz × 4)
OS 用ストレージ	Samsung SSD PM863a 480 GB (SATA3.0)
DB 用ストレージ	Intel SSD DC P3500 2 TB (PCIe)
電力計測器	Yokogawa WT1800

表 2: システムのソフトウェア構成．

OS	CentOS 7.5.1804 (Linux kernel 3.10.0)
DBMS	PostgreSQL 9.6.8
データセット	TPC-H 2.17.3

表 3: PostgreSQL のメモリ割り当て関連の設定．

`shared_buffers`	8192 MB
`temp_buffers`	2048 MB
`work_mem`	4096 MB / M （M はワークロードの多重度）

3 実験設定

本節では 2.3 節で記述したワークロードの実現のために行った，オープンソースのデータベースシステムを利用した実験に関して説明する．

3.1 測定環境

サーバのハードウェア構成を表 1 に示す．プロセッサとして 18 コア CPU を 2 ノード搭載しており，計 36 の物理コアが利用できる．SATA3.0 で接続された内蔵 SSD を OS 用ストレージとして利用し，それとは別に外付け SSD を PCI Express で接続して DB 用ストレージとした．このようにデータベースシステムが管理するデータを DB 用ストレージ上に置くことで，データベースに対する IO は他と区別可能にした．そして，このサーバの給電系統を電流・電圧測定用回路を経由させて接続し測定用回路から高精度電力計 Yokogawa WT1800 へ信号を入力することで，サーバ全体の消費電力を測定可能にした．

次にソフトウェア構成を表 2 に示す．DVFS 機能には `cpufreq` と呼ばれるカーネルモジュールを利用しており，今回利用した CPU においては各コアに対して 1.0 GHz から 2.3 GHz の間で

0.1 GHz 刻みで動作周波数を設定できる．本実験では動作周波数を 1.0 GHz に固定した．また，プロセッサにはアイドル時に C-State と呼ばれる状態に移行して電力消費を抑える機能が搭載されているが，そのうちの最も省電力性が高く復帰時間が長い C6-State が有効な環境下では消費電力計測の上で挙動の把握に困難があったため，これを無効にして実験を行った．PostgreSQL [13] が利用する共有メモリサイズ等の設定は表 3 に示すように設定した．ここで，work_mem に関してはワークロードの多重度 M に応じて設定するようにした．TPC-H は公式で提供されている dbgen ツールを用いてスケールファクタ 300 のデータベースデータと問合せを生成した．

サーバのシステム負荷の実行時計測としては，sysstat パッケージが提供する sar コマンドにより各コアの CPU 利用率を，diskstats により DB 用ストレージの IO 統計情報を，ipmitool コマンドによりプロセッサの温度と対応するファン速度を，それぞれ 1 Hz で取得した．また，消費電力は WT1800 との通信により 20 Hz で取得した．

3.2 実験用ワークロード

実験におけるワークロードの設定について説明する．本実験では 2.3 節において議論したように，データベースへの走査範囲の広さを変化させることで必要となる IO サイズを変え，資源律速が切り替る様子を把握することを目的としている．

a） 定常的なワークロード.

定常的なワークロードを考えるために，ある一定の多重度で問合せ処理をさせ，各プロセスの問合せの完了回数が閾値 I を超えるまで継続することとした．ワークロードの終了時点までは全プロセスが問合せ処理を中断せず，次々と問合せ処理を行う．以下，プロセスが処理する対象の問合せが入ったキューを「問合せキュー」と呼ぶ.

b） 利用した問合せ.

問合せには TPC-H の q3 に対して改変を加えた q3’ を利用した．具体的には，dbgen で生成した q3 の where 節に対して，CUSTOMER 表の主キーである c_custkey の範囲によって絞り込む条件を追加した．本実験では c_custkey が等分割されるように範囲を規定している．スケールファクタ 300 の場合，c_custkey は総数 4500 万であるので，これを N 分割するためには $K = 45 \times 10^6/N$ 個ずつ区切って範囲を定めることになる．こうして得られた N 個の範囲をランダムに並び替え，dbgen によって生成された N 個の q3 に対しそれぞれ適用した．作成した q3’ の例としてリスト 1 にソースコードを示す.

c） 問合せキューの割り当て.

作成された q3’ は N 種類存在するが，このうちの n 種類のみを利用し，多重度 M で処理することを考える．つまり，$n = 1$ の時は全プロセスにおいて同一の問合せを処理する．この場合，n が 1 に近づくほどデータベースへのアクセス範囲は狭くなり，n が M に近づくほどデータベースへのアクセス範囲は広くなることが予想される．再現可能なワークロードにするため，どのプロセスがどの問合せを利用するかは決定的に割り当てることとした．つまり，各プロセスが一つずつある決まった問合せキューを持つ．割り当て方としては，n 種類の問合せを一種類ずつ順番に M 個のプロセスに割り当てていく．それぞれのプロセスに最低でも一種類の q3’ を割り当てたいので，$n < M$ の場合は途中で n 種類の問合せを使い切った時は再補填して割り当てを再開し，最後のプロセスに問合せが割り当てられたら終了とする．$n \geqq M$ では n 種類の問合せが全て割り当てを完了した時点で終了とする．こうして割り当てられた問合せを問合せキューの中身とする．なお，問合せキューは循環するものとする.

リスト 1: 生成された q3’ の SQL 例．下線部が，元となる q3 の SQL に追加した部分．$K = 45 \times 10^6/N$ かつ l は 0 以上 N 未満の整数.

```
1  -- using 1525505385 as a seed to the RNG
2
3
4      select
5          l_orderkey,
6          sum(l_extendedprice * (1 - l\_discount)) as
               revenue,
7          o_orderdate,
8          o_shippriority
9      from
10         customer,
11         orders,
12         lineitem
13     where
14         c_mktsegment = 'AUTOMOBILE'
15         and c_custkey between Kl+1 and K(l+1)
16         and c_custkey = o_custkey
17         and l_orderkey = o_orderkey
18         and o_orderdate < date '1995-03-20'
19         and l_shipdate > date '1995-03-20'
20     group by
21         l_orderkey,
22         o_orderdate,
23         o_shippriority
24     order by
25         revenue desc,
26         o_orderdate;
```

d） ワークロード予備動作.

連続して問合せ処理を行うワークロードを定常的にするためには，コールドスタートの影響を取り除く必要がある．そのために，予備動作として M 個のプロセスに問合せキューの 1 番目の問合せをそれぞれ処理させてからワークロードを開始させることとした．つまり，ワークロードの計測開始は問合せキューの 2 番目の問合せの処理からとなる.

e） パラメータ.

本実験では多重度 $M = 360$，q3’ の総種類数 $N = 3600$，ワークロード終了のための問合せ処理完了回数の閾値 $I = 4$ とした．ワークロード中で利用する q3’ の種類数は，$n = 1, 5, 10, 15, 20, 25, 30, 40, 50, 100, 200, 400, 1200, 3600$ を選択し，計測における分散の存在を考慮して各条件で 3 回ずつワークロードの計測を行った．多重度 360 のプロセスを処理す

るために利用する CPU コアは，2 ノードある CPU のうちの片方 18 コアに均等に割り振った．

3.3 追加実験

前節で述べた条件での主実験に加え，サーバ設定による影響を確認するために行った追加実験に関して述べる．

a） Hyper-Threading（HT）の有無による影響

実験で利用したサーバのプロセッサは Intel の Hyper-Threading（以下，HT）と呼ばれるマルチスレッディング機能を備えており，1 コアあたり 2 スレッド利用できるため論理 CPU としては 72 コアと認識される．HT 機能により異なるパイプラインを同時利用することが可能になるため，プロセッサのリソースを有効活用に繋がるとされている．本研究では基本的に HT を有効にして実験を行ったが，HT による影響を検証するため HT を無効にした条件下での実験も行った．

b） キャッシュミス率の監視

近年のコンピュータでは，OS が提供するソフトウェアカウンタや PMU 等のハードウェアカウンタを通じて，キャッシュミス率等の統計情報を得ることができる．実験では `perf` と呼ばれるコマンドを通じてワークロード中のキャッシュミス率を取得したが，プロセッサへの負荷が無視できない程度に存在する．そのため，`perf` を利用した統計情報の取得は追加実験として主実験とは別に行い，比較を行った．

4 実験結果とその考察

4.1 主実験と HT 無効の追加実験の結果

3 節で記述した，問合せの種類数を変化させることでデータベース走査範囲を変化させるワークロードの実験結果を図 1 に示す．各種類数の条件を 3 回ずつ計測しているため，これらの平均を取っている．また，同時に 3.3 節で記述した HT 無効の条件における結果に関してもプロットしている．図 1a は実験で利用した全ての種類数の条件に関してプロットしており，図 1b は問合せ種類数が 400 までの部分に限定したものである．

グラフにより HT による影響はほとんど無視できる程度であることが分かる．これは，今回ランダム IO が主体となるワークロードであったために，パイプラインの並列化が効果的になるような状況ではなかったことを示唆している．

全体的な傾向としては，問合せの種類数が増えるほど CPU 利用率が徐々に下がっていく様子が分かる．これは CPU 律速から IO 律速に変化していっている様子を示している．CPU 利用率の推移の仕方を詳しく見ると，範囲によって次のように区分できる．

(i) 種類数 1〜20: CPU 利用率が 1800 % でほぼ一定

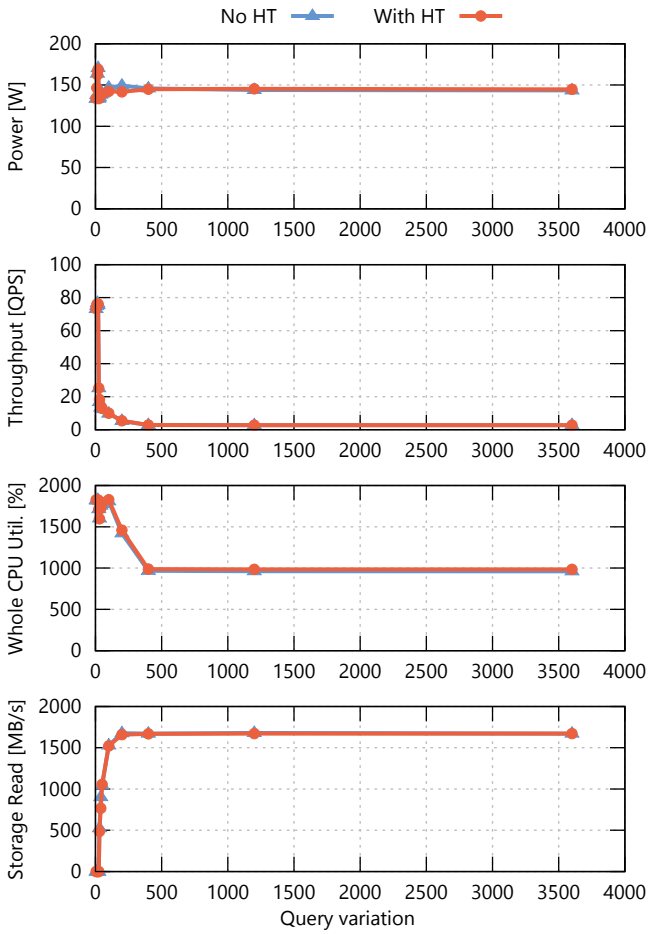
(ii) 種類数 20〜100: 種類数 30 までは CPU 利用率が 1500 % 近くまで下落した後，再び 1800 % に戻る

(iii) 種類数 100〜3600: CPU 利用率が下落し 1000 % 前後で推移

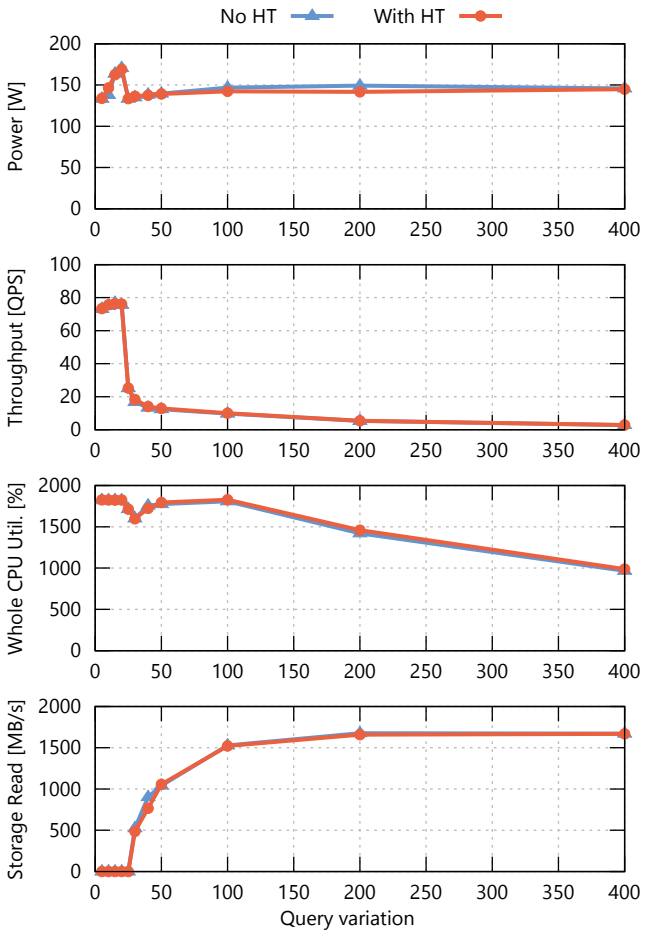
(i) の区間においては，完全に CPU 律速になっており負荷をかけている 18 コア全てで CPU 利用率がほぼ 100 % となっていた．ストレージ IO スループットを見ると約 0.014 MB/ sec と非常に小さい．これは，ワークロード予備動作により必要なデータベース走査範囲がほぼ全てメモリ上に収まっており，ワークロード実行中にはストレージ IO が不要となるためである．消費電力は，ストレージ IO がなく CPU 利用率もほぼ一定なため変化は大きくないことを予想していたが，実際には種類数が増えるほど上がった．ここで，(i) の区間に属する条件のうち，種類数 1, 10, 20 の経時変化をプロットしたものを図 2 に示す．種類数が上がった際には消費電力が途中から高くなるという挙動になっている．現状では消費電力が上昇する理由を正確に把握するには至っていないが，理由として考えられるのが，キャッシュミス率が増加することによってキャッシュあるいはメモリにおいて高頻度な置換が発生しているという可能性が挙げられる．

(ii) の区間においては，CPU 利用率が谷を描くような挙動が観察される．この区間は CPU 律速と IO 律速の間の中間領域であると見ることができる．(i) の区間と比べるとスループットが急激に下がっていると同時に，消費電力も急激に下がってから徐々に上がっている．このように中間領域では特異な性質が見られる．種類数 25, 30, 50 の経時変化をプロットしたものを図 3 に示す．種類数 25 は，消費電力が 130 W 前後，CPU 利用率が 1700 % 前後でほぼ一定で推移している．そしてストレージ IO がほとんど発生していないにも関わらず，種類数 20 に比べて急激に問合せ処理スループットが下がっている．これは，システムによるキャッシュの退避が行われたことによって，データが OS 用ストレージ上のシステムキャッシュに存在していたからだと考えられる．つまり種類数 20 まではデータアクセスがメインメモリまでに留まっていたが，種類数 25 ではストレージ上のキャッシュを利用する必要がある程度にまでアクセスサイズが増加したと見なせる．種類数 50 においても消費電力，CPU 利用率，ストレージ IO スループットが共にほぼ一定で推移している．つまりほぼ定常のワークロードとなっている．対して種類数 30 は，初めは消費電力が 130 W で推移しているが 60 秒付近から大きく振れるようになっており，CPU 利用率を見ても 1700 % で推移していたが急激に下がってから時間と共に上昇していくという変化の仕方をしている．CPU 利用率が急激に下がるポイントでは，ストレージ IO スループットも下がっていることから，データの取り方がそれまでとは異なる手法に切り替わっている可能性がある．現時点ではこの理由に関しては，分析が不十分となっている．

最後に，(iii) の区間は IO 律速となっており，CPU 利用率や問合せ処理スループットは下がっていく一方で消費電力はほとんど変わっていない．更に種類数 400 以上の範囲になるとグラフの数値上での違いがほとんどなくなっている．これは，多重度が 360 であるため問合せ種類数をそれ以上に設定しても同時に走査する範囲は増加しないためである．種類数 100, 200, 400 の経時変化をプロットしたものを図 4 に示すが，定常的なワークロードとなっていることが分かる．



(a) 問合せの種類数: 1〜3600



(b) 問合せの種類数: 1〜400

図 1: 問合せ種類数を変化させた時のワークロードの変化．上から消費電力，問合せ処理スループット，全コアの CPU 利用率，ストレージ IO スループット．

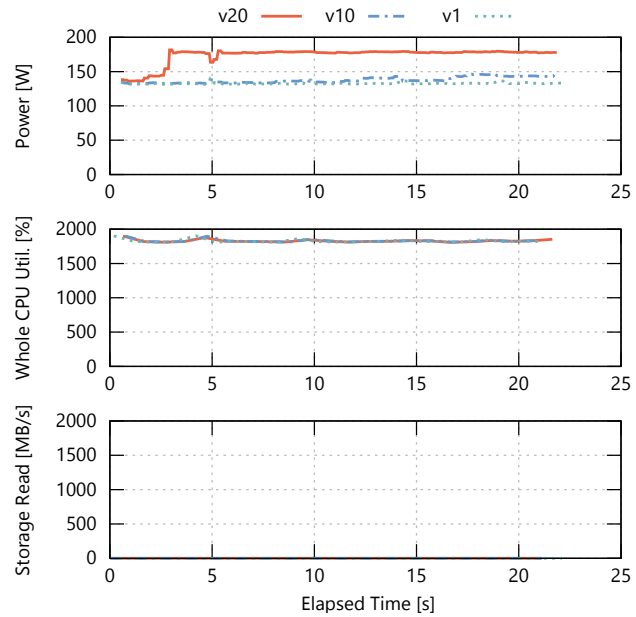


図 2: 種類数 1, 10, 20 におけるワークロード中の消費電力，全コアの CPU 利用率，ストレージ IO スループットの経時変化．

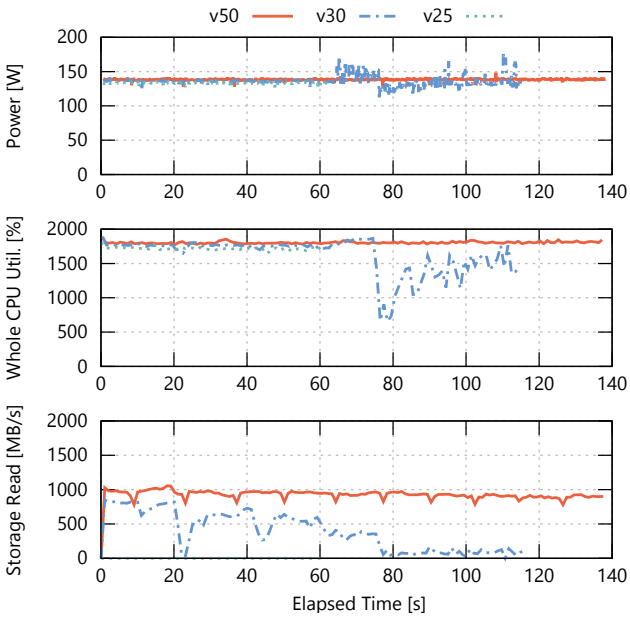


図 3: 種類数 25, 30, 50 におけるワークロード中の消費電力，全コアの CPU 利用率，ストレージ IO スループットの経時変化．

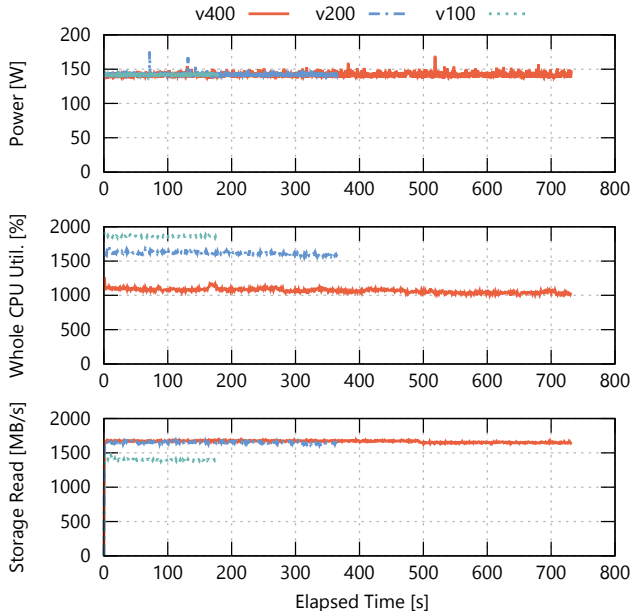


図 4: 種類数 100, 200, 400 におけるワークロード中の消費電力，全コアの CPU 利用率，ストレージ IO スループットの経時変化．

4.2 キャッシュミス率の監視を行った追加実験の結果

`perf` を利用してキャッシュミス率を監視を行った際の，各問合せの種類数での実験結果を図 5 に示す．ここでは種類数 1〜400 までの結果のみ載せており，主実験の結果との比較用のグラフもプロットしている．CPU 利用率が主実験の時よりも上がっていることが確認できるが，それに伴い問合せ処理スループットは低くなっている．

キャッシュミス率に関して見てみると，4.1 節で行った区間分けと同期するような挙動になっている．(i) の区間ではミス率が上昇していくが，これは利用するメインメモリの範囲が広がっていくことから理解される．やはりキャッシュミス率の上昇が消費電力の上昇と関わりがあると考えられる．そして (ii) の区間ではミス率が下がり，(iii) の区間ではミス率が上がっていく．種類数 25 でミス率が急激に下がっているのは，ストレージ上のキャッシュまでアクセスする場合はキャッシュミスとして扱われなくなるからだと考えられる．(ii) と (iii) の区間におけるキャッシュミス率と消費電力・CPU 利用率との関係性については更なる検証が必要である．

5 おわりに

本論文では，分析問合せ処理における資源律速に着目し，その変化による影響を独自のワークロード設定により確かめた．CPU 律速となるか IO 律速となるかはデータベースのアクセス範囲に大きく依存していることを確認した他，定常的なワークロードであるという仮定が適切でない場面も確認できた．特に CPU 利用率やストレージ IO スループットからは直接判断できない消費電力増加現象，資源律速変化の境界における CPU 利用率が急激に下落する現象などは，キャッシュミス率との関連性がある可能性を提示した．

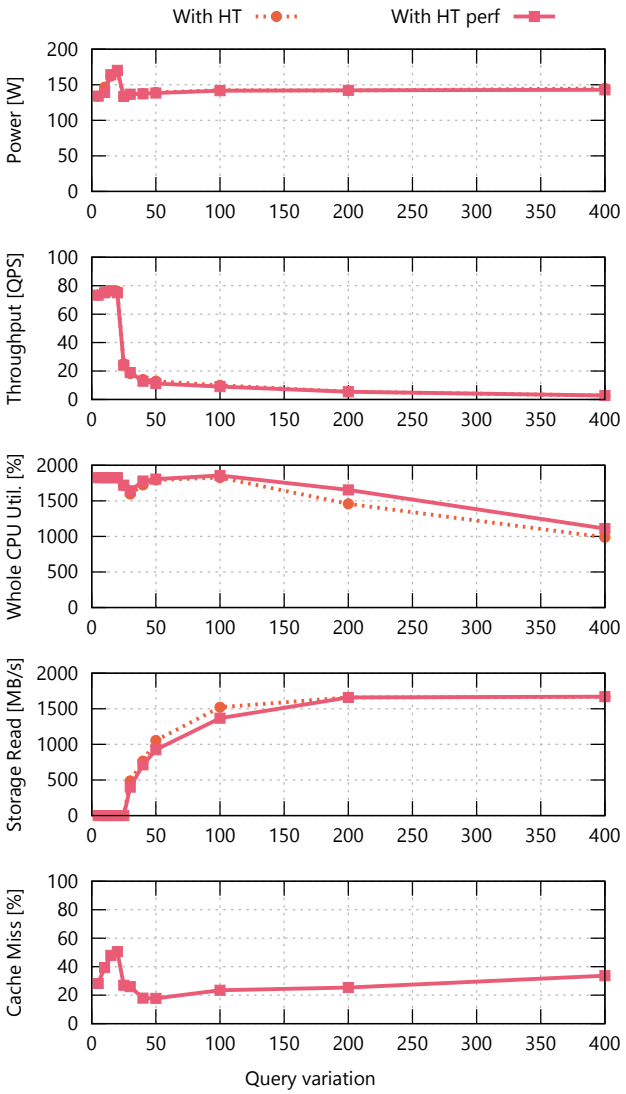


図 5: 問合せ種類数を 1 から 400 まで変化させた時のワークロードの変化．上から消費電力，問合せ処理スループット，全コアの CPU 利用率，ストレージ IO スループット，処理全体におけるキャッシュミス率．

今後の課題として，現状では挙動の理由の説明を十分に行えていないため，それに向けた更なる実験を行いたい．例として，メモリアクセス情報やプロセッサアンコアの統計情報を取得することでより多方面から材料を得ることが挙げられる．また，細かい粒度で現象を捉えるために今回の実験で利用したものよりも簡易なワークロードを考案し，細部の挙動の分析を繋げることが考えられる．

文　献

[1] Josh Whitney and Pierre Delforge. Data Center Efficiency Assessment. Issue paper, 2014.

[2] Paolo Bertoldi, Nicola Labanca, and Bettina Hirl. *Energy efficiency status report 2012: electricity consumption and efficiency trends in the EU-27.* Publications Office, 2012.

[3] Willis Lang, Ramakrishnan Kandhan, and Jignesh M. Patel. Rethinking Query Processing for Energy Efficiency: Slowing Down to Win the Race. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering (TCDE)*, Vol. 34, No. 1, pp. 12–23, March 2011.

[4] Mayuresh Kunjir, Puneet K. Birwa, and Jayant R. Haritsa. Peak Power Plays in Database Engines. In *Proceedings of the 15th International Conference on Extending Database Technology (EDBT)*, pp. 444–455, 2012.

[5] Zichen Xu, Yi-Cheng Tu, and Xiaorui Wang. PET: Reducing database energy cost via query optimization. *Proceedings of the VLDB Endowment*, Vol. 5, No. 12, pp. 1954–1957, 2012.

[6] Amine Roukh, Ladjel Bellatreche, and Carlos Ordonez. EnerQuery: Energy-Aware Query Processing. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM)*, pp. 2465–2468, 2016.

[7] 早水悠登, 合田和生, 喜連川優. ストレージ消費電力特性に基づく関係データベース演算子の省電力指向コストモデル. 第 9 回データ工学と情報マネジメントに関するフォーラム (DEIM), March 2017.

[8] 羅博明, 早水悠登, 合田和生, 喜連川優. プロセッサ動作モード制御による分析指向問合せ処理の省電力化効果の測定. 第 10 回データ工学と情報マネジメントに関するフォーラム (DEIM), 2018.

[9] Boming Luo, Yuto Hayamizu, Kazuo Goda, and Masaru Kitsuregawa. Modeling Query Energy Costs in Analytical Database Systems with Processor Speed Scaling. In *Database and Expert Systems Applications*, Vol. 11030, pp. 310–317. Springer International Publishing, 2018.

[10] Transaction Processing Performance Council. The TPC Benchmark H (TPC-H). `http://www.tpc.org/tpch/`.

[11] Meikel Poess, Bryan Smith, Lubor Kollar, and Paul Larson. TPC-DS, Taking Decision Support Benchmarking to the Next Level. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pp. 582–587, 2002.

[12] Pat O'Neil, Betty O'Neil, and Xuedong Chen. Star Schema Benchmark Revise 3. June 2009.

[13] The PostgreSQL Global Development Group. PostgreSQL: The world's most advanced open source database. `https://www.postgresql.org/`.