修士論文　　2009年度（平成21年度）

実空間コンテキストに応じたコンテンツ配信

慶應義塾大学大学院 政策・メディア研究科
佐藤　龍

修士論文要旨　　2009 年度（平成 21 年度）

実空間コンテキストに応じたコンテンツ配信

論文要旨

近年，掲示板などの情報表示媒体はデジタルサイネージ端末としてデジタル化されつつある．これに伴い，複数の情報を切り替えて表示できるデジタルサイネージ端末を用いて，情報閲覧者に応じた情報を配信する活動が盛んに行われている．配信時の気候や配信場所などの実空間コンテキストに応じて配信する情報を変えるダイナミックデジタルサイネージは，特定の閲覧者に対する情報の配信や，時間が経つことで価値が損なわれる情報を即座に配信することができると期待されている．一般に，実空間コンテキストは実世界における事象の変化と同程度の頻度で変化が観測される．高い頻度で変化する実空間コンテキストは物や人の移動であるため，デジタルサイネージ端末が移動する場合は，より多くの情報を処理する必要がある．また，移動に伴う通信環境の変化により配信に必要な時間が不安定になる問題を解決する必要がある．

本論文では，実空間コンテキストをいくつかの場合に分類するための条件が与えられているとき，観測した実空間コンテキストがいずれの場合に属するかを，条件間の関係により効率的に検索できる手法を実現した．また，検索の結果より得られた場合に応じた情報を，移動するデジタルサイネージ端末へ安定して配信する手法を実現した．本研究により，移動体を対象とした実空間コンテキストに応じたコンテンツ配信が可能となり，実空間コンテキスト検索の高速化と，移動端末への情報配信の安定化に貢献できる．

キーワード
1．実空間コンテキスト，2．モバイル，3．デジタルサイネージ，4．非同期，
5．プリフェッチ，6．インデクシング

慶應義塾大学大学院 政策・メディア研究科
佐藤 龍

Abstract of Master's Thesis

Academic Year 2009

Real-Space Context Aware Content Devlivery

Summary:

Recently, the medium of the display of information such as bulletin boards is being digitalized as a digital signage device. There are some experiments to delivery content adjust for viewer by switching between several contents. It is challenging to keep quality of the content of a dynamic digital signage (which changes content according to the real-space context of delivery viewer's location, weather, etc.) since the quality will be degraded as time passes. Real-space context was generally sensed with mostly same duration as change of state. One of the most frequently changed real-space context is movement of people and vehicle. Therefore, it is necessary to process large number of contexts in the case where digital signage device installed on a vehicle. Stabilization of delivery time is also important.

In this thesis, design and implementation of efficient search method to classify real-space context by pre-configured conditions with help of their relation, and a method to stabilize delivery of contents for mobile digital signage device will be discussed. The results of this study contribute efficient search of real-space context search and stabilize deliver contents for mobile devices.

Keywords:
1. Real-Space Context, 2. Mobile, 3. Digital Signage, 4. Asynchronous, 5. Prefetching, 6. Indexing

Keio University, Graduate School of Media and Governance
Ryu Sato

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景

インターネットでは様々な実空間の情報を利用したサービスが広く提供され始めている．従来，実空間情報は実空間における様々な状態を示す情報として利用されており，インターネットの普及により活用の場が広がっている．天気予報では，大気の温度や圧力，湿度，照度などの実空間情報を統合して活用し，カーナビゲーションでは，車の位置や，速度，進行方向などの実空間情報を統合活用してきた．今日，世界的な情報通信インフラであるインターネットにより，逐次的に観測した実空間情報を伝えられるようになったため，リアルタイムに情報を提示するサービスが提供されつつある [1, 2]．実空間情報をリアルタイムに用いることにより，従来では実現できなかった有益なサービスが提供され始めている．

一方，無線技術の発展に伴い，屋外においてもラップトップ PC や PDA などの携帯端末を用いてインターネットに接続できるようになった．インターネットを通じて複数の情報端末が連携動作することにより，デジタルサイネージなどの情報表示システムにおいて，常に新しい情報を表示することが可能である．近年，時間経過に応じて価値が下がる情報に対し，即時性を高めることで価値の低下に対応するシステムの実現が期待されている．また，情報の配信先が車内の情報端末である場合において，緊急性や即時性を有する情報は通信規格が対応していないため配信が困難であった．近年実用化が進んでいる WiMAX は，通信品質を調節可能な広域無線通信規格として注目されている．従来は車などの移動体における情報の表示方法は，紙による表示方法や，端末に蓄積された情報をディスプレイにより表示する方法であったが，WiMAX などの広域無線環境を利用することで，時々変化する情報を表示できるようになる．

時々変化する情報の一つに Web コンテンツがある．時々変化する Web コンテンツをデジタルサイネージを用いて配信することにより，閲覧者に価値のある情報を配信することが可能である．例えば，鉄道会社やバス会社などは障害時に障害が運行に与える影響を Web で公開している．運行情報は新しいことに価値があり，古い情報は移動時間を浪費するなどの悪影響を及ぼすため即時性が必要である．また最近では，CGM(Consumer Generated Media)（特に商店や自治体のミニブログやそれらの RSS 情報）など，比較的頻度高く更新される Web コンテンツが増加している．運行情報や CGM など，更新頻度が高いコンテンツは即時性が要求される Web の情報である．また，交通情報は即時性を求める一方で，利用しない交通機関の情報には価値が少ないため，個人が情報を取得するのが現状である．デジタルサイネージは表示する情報を変化させることが可能であり，閲覧者が閲覧する地域が設置する場所に限定されるた

め，即時性のある Web コンテンツを提供するのに適している．

実空間情報を用いて即時性の高い Web コンテンツなどの情報を配信することにより，閲覧者に価値のある情報を提供できると考えられる．実空間情報を活用するサービスには，リアルタイムに変化する実空間情報を逐次処理するための即時性が必要である．また，将来的には対象とする移動体の数が増加する事を考慮すると，車や電車などの移動体の位置を識別する規模性を持つ事が望ましい．本研究は，移動体を対象として，実空間情報に応じた情報配信を行うためのシステムの実現を目指す．

1.2 本研究の目的

本研究の目的は，移動体の位置とセンサが観測した情報（センシングデータ）に応じて，情報を配信するシステムの実現である．現状の問題を解決するとともに，将来の展望を見据え，複数の情報端末への情報配信ができ，多様なセンシングデータを効率的に検索できるシステムの実現を目指す．

1.3 本論文の構成

本論文は 9 章から構成される．第 2 章では，実空間コンテキストに応じた情報配信システムの概要について述べ，筆者らの活動から得られた知見より，システムに必要な要件を述べる．第 3 章では，システムを実現する上での問題点を述べる．第 4 章では，関連研究について述べる．第 5 章では，問題を解決するためのアプローチを示し，第 6 章でシステムの設計を述べた後，第 7 章で実装の詳細を述べる．第 8 章で評価を行い，第 9 章で結論を述べる．

第2章　実空間コンテキストに応じたコンテンツ配信

近年，コンテンツ配信システムにおいて，屋外や店頭などに設置したディスプレイやプロジェクタを用いて情報を表示するデジタルサイネージが注目されている．デジタルサイネージを用いた活動では，情報の閲覧者からインタラクティブな操作を受け付け，表示端末が移動環境に設置されていることを想定するなど，実空間コンテキストに応じた情報配信が求められている．筆者らは慶應義塾大学湘南藤沢キャンパスの巡回バス内にデジタルサイネージシステムを構築し研究を行っている．本章では，デジタルサイネージによるコンテンツ配信システムについて述べ，筆者らが構築したデジタルサイネージシステムについて述べる．

2.1　デジタルサイネージ

デジタルサイネージ [3] は，ディスプレイやプロジェクタに映像や広告などの情報をデジタル技術を用いて表示する端末である．紙での表示に対して，比較的表示する内容を変更しやすい．また，一つの表示端末で動画など内容が変化する情報を表示できるなど，紙と比較して多くの種類の情報を表示することができる．

近年では，デジタルサイネージを利用した活動が広く行われている．Station Vision[4] は東京メトロ丸ノ内線の主要 6 駅（東京駅，銀座駅，赤坂見附駅，新宿駅，新宿三丁目駅，中野坂上駅）のホームで，72 台のディスプレイに広告配信を開始する予定のデジタルサイネージである．Station Vision は液晶ディスプレイとスピーカーにより，複数の音声付映像広告を，決められた時間枠の間に放映する．メディアポール [5] は，韓国ソウル市とカンナム市の交差点の通り沿いに設置しているデジタルサイネージである．メディアポールは液晶ディスプレイにより表示を行うとともに，タッチパネルにより閲覧者からのインタラクティブな操作を受け付ける．

デジタルサイネージは複数の情報を表示することが可能な表示端末であり，閲覧者から情報選択のための情報を受けるものも存在する．情報選択のための情報を受ける手段は閲覧者からの直接操作を受け付ける方法やセンサによる観測情報を取得する方法が用いられる．慶應義塾大学 SFC 研究所アンワイヤード研究コンソーシアムは NTT レゾナントと共に，デジタルサイネージにおける表示情報を選択するための情報をセンサを用いて自動で取得する実験を行った（図 2.1）．デジタルサイネージは複数の表示可能な情報から，閲覧者に適した情報を選択して表示できる技術として注目されている．

(a) ディスプレイ　　(b) センサデバイスと取り付けの様子

図 2.1: 慶應義塾大学で行われたデジタルサイネージ実験の様子

2.2 移動デジタルサイネージ

情報表示端末としてデジタルサイネージが設置される場は固定された場所に限らない．車や電車などの車内にデジタルサイネージを設置する活動がある．本論文では，デジタルサイネージを用いる情報配信において，車や電車などの移動体に表示端末が設置される場合を**移動デジタルサイネージ**と呼ぶ．移動デジタルサイネージにおいて，移動する表示端末への情報配信は無線技術を用いて行う点が特徴である．

近年，電車やバスの車内での広告配信に移動デジタルサイネージが用いられ始めている．JR 東日本が運営するトレインチャンネル [6] は，4 つの路線（山手線，中央線快速・青梅五日市線，京浜東北線，根岸線）で車両ドア上に設置された液晶ディスプレイに，ニュース・天気予報・占いなどの情報番組を放映する移動デジタルサイネージである．トレインチャンネルは三菱電機のトレインビジョンシステム [7] を用いている．トレインビジョンシステムは大容量動画のデータ転送にミリ波を用いた車両内放映システムである．また，TOKYO FM は西日本鉄道などと共同で，西鉄バス車内に読売新聞のニュースや福岡市の観光，行政情報などを配信する移動デジタルサイネージ実験を開始した [8]．

移動デジタルサイネージは，移動する表示端末においてもデジタルサイネージと同様に，複数の情報を閲覧者に効果的に表示できると期待されている．

2.3 慶應義塾大学キャンパス内の移動デジタルサイネージ

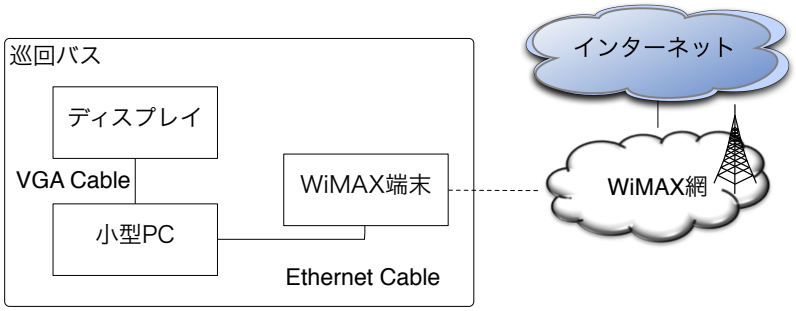
筆者らは慶應義塾大学湘南藤沢キャンパスにおいて，学生を対象とした巡回バス車内に移動デジタルサイネージシステムを構築した（図 2.2）．図 2.2 のシステムは，バス車内に設置した液晶ディスプレイに交通情報やブログの情報など，複数の Web ページを切り替えながら表示する．表示する Web コンテンツは小型 PC 上のアプリケーションが WiMAX 網を通じて取得する．

筆者らは，図 2.2 に示した移動デジタルサイネージシステムにおいて，位置に応じた情報の選択や，閲覧者の人数などに応じた表示情報の選択など，実空間コンテキスト

(a) キャンパス巡回バス（SoKanKan）外観

(b) キャンパス巡回バス（SoKanKan）内観

(c) WiMAX 端末（CPE）



(d) システム構成図

図 2.2: 慶応義塾大学キャンパス内の移動デジタルサイネージシステム

に応じた情報を選択すべく計画している（2.3）．森川ら [9] は実空間情報を用いた実空間指向のサービスにおけるコンテキストを，ユーザコンテキスト（ユーザプロファイル，位置，行動など），物理コンテキスト（明るさ，雑音，交通状態，温度など），コンピューティングコンテキスト（ネットワーク接続性，通信コスト，近隣デバイスなど），時間コンテキストなどの多様な意味を示すと定義している．本論文において，森川らのコンテキストの定義を本研究の対象に適応させ，**実空間コンテキスト**を，ユーザコンテキストと時間コンテキスト，物理コンテキストを示すと定める．また，ユーザコンテキストにおけるユーザを移動端末の閲覧者と定める．

移動デジタルサイネージシステムにおいて，位置情報や交通情報など物理コンテキストや時間コンテキストが必要である．例えば，昼に商店街をタクシーやバスが通ると，飲食店の昼食情報を提示するサービスや，店舗の周辺に限定したタイムセール情報の提示するサービス，また，突然発生した事故により電車の運行が停止した場合に，交通障害情報に加えて代替経路の情報を提示するサービスにはこれらの情報が必要である．実空間コンテキストに応じて情報を表示することより，従来のように，固定された情報を表示するシステムでは実現できなかったサービスが提供できる．本研究は，移動デジタルサイネージにおいて，実空間コンテキストに応じて情報を配信すること

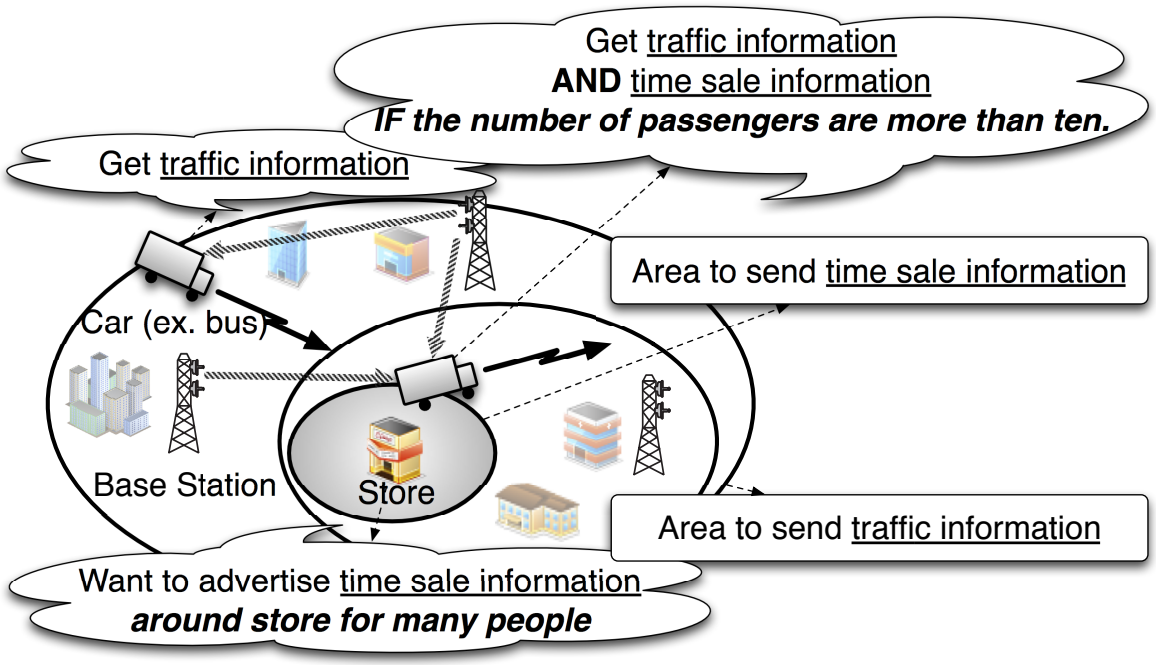


図 2.3: 実空間コンテキストに応じたコンテンツ配信の概要

を目的とする．

2.4 まとめ

コンテンツ配信システムにおいて，デジタルサイネージを用いる活動が普及していることを述べた．デジタルサイネージでは閲覧者に適した情報を配信するため，実空間コンテキストに応じて表示する情報を選択することが求められている．本研究は，情報表示にデジタルサイネージを用いるシステムにおいて，**移動体への情報配信の考慮**と**実空間コンテキストに応じた情報配信**を実現することを目標とする．

第3章　実空間コンテキストに応じたコンテンツ配信における課題

本章では，実空間コンテキストに応じたコンテンツ配信システムにおける課題について議論し，本研究の目標を実現するための要件として整理する．移動デジタルサイネージシステムは，移動体の位置情報などの実空間コンテキストを扱う必要があることを第2章で述べた．実空間コンテキストに応じたコンテンツ配信システムの要件を整理するため，複数の移動体を考慮し，システムにおいて必要となる情報について機能面での性質より議論する．複数の移動体を考慮することにより，一般的な実空間コンテキストに応じたコンテンツ配信システムにおける問題点を整理できる．

3.1　実空間コンテキストに応じたコンテンツ配信の特徴

実空間コンテキストに応じてコンテンツを配信するためには，配信元から配信先へコンテンツが配信される過程で，実空間コンテキストの解析が必要となる．実空間コンテキストの解析とは，起こると想定される実空間コンテキスト（**実空間コンテキスト条件**）と実空間コンテキストの観測結果が一致するかどうかを判別する処理である．図3.1に，実空間コンテキストに応じたコンテンツ配信システムにおける，コンテンツ配信元と配信先，実空間コンテキストの解析処理の関係と，システムが扱う必要のある情報の性質を示す．コンテンツ配信元は配信する情報を指定するシステム利用者である．コンテンツ配信先は移動体に設置された情報表示端末である．

実空間コンテキストは，ユーザコンテキストと物理コンテキスト，時間コンテキストを示す．ユーザは移動体に設置された情報表示端末の閲覧者であり，ユーザコンテキストにおける位置情報はコンテンツ配信先の移動と共に変化する．また，物理コンテキストと時間コンテキストも同様に時々刻々と変化する．従って，実空間コンテキストは頻度高く変化する特徴がある．一方，配信する情報に関連する実空間コンテキスト条件や配信する情報も動的に変更される場合がある．実空間コンテキスト条件とは情報を配信するための条件を示す．例えば，場所を限定したタイムセールの配信を考えると，セールする商品の内容を変える場合や，配信対象者をセールする商品によって変更する場合が考えられるため，動的に変更される場合がある．実空間コンテキストに応じた情報配信システムが扱う情報は，まとめると次の3点に示す特徴がある．

- 実空間コンテキストは頻度高く変化する
- 実空間コンテキスト条件は配信元の指示により動的に変更されうる

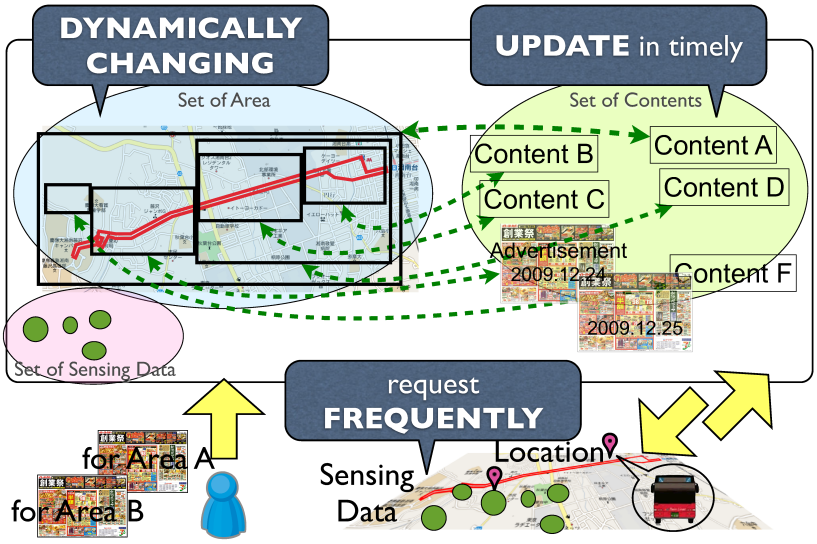


図 3.1: 情報配信システムが扱う情報取得環境の関係と性質

- 配信する情報は配信元により更新されうる

3.2 即時性のある情報への対応

本研究では，配信する情報は交通情報やブログなど Web コンテンツを想定する．交通情報などは，情報が価値を持つ時間が比較的短く，時間が経つことにより有益な情報が不要な情報に変化するため即時性を求める．さらに，バスやタクシーなどは移動するため，位置情報は頻繁に変化する．これらの情報を扱うには即時性が必要であり，実空間コンテキストに応じたコンテンツ配信システムが備えるべき要件と言える．上述の交通情報やブログ（特にミニブログ）のようなリアルタイム性を要求される Web 情報では，ネットワークのキャッシュサーバ間を移動する時間間隔よりも短い間隔で，最新の情報を取得できることが望ましい．以上の議論より，実空間コンテキストに応じた情報配信システムは即時性を持つ必要がある．

3.3 規模性を持つ情報処理性能の考慮

実空間コンテキストに応じたコンテンツ配信を行うためには，実空間コンテキストを頻度高く解析できる処理能力が必要である．実空間コンテキストを解析するには複数の移動体と複数のセンサノードからのセンシングデータを処理する必要がある．一つのセンシングデータはデータサイズが小さいが，対象となる地理範囲が広くなるに従い，網羅するためにセンサ数を増やす必要があるため，処理する必要のあるデータ数は増加する．例えば，東京都と神奈川県を例とする 1 都道府県，または藤沢 WiMAX のサービス範囲を網羅するシステムを考え，40bit のセンシングデータが 1s 間隔でデータ

を送信した場合のデータ量を計算した結果を，表 3.1 に示す（詳細な動作設定は A.1 を参照）．ここで，GPS など位置情報を緯度経度により表す場合，一般的におよそ 40bit 精度，1s 間隔で測定されるため，この値を用いて処理する必要のあるデータ数を予想した．結果，システムが対象とするデータ数は千から数十万となり，膨大なデータへ対応する必要性があると分かる．現状において多数の情報端末を扱える事はシステムとしての利点であるとともに，膨大なデータ数を検索するに耐えうるシステムが，将来的には求められると予想できるため，膨大なデータ数への対応は実空間コンテキストに応じたコンテンツ配信システムにおける課題の一つである．

	データ数/秒	平均ビットレート (Mbps)
東京都	210300	3.1
神奈川県	241600	3.6
藤沢 WiMAX	1256	0.6

表 3.1: 予測されるセンシングデータの処理性能

対象とする地理範囲の拡大に伴い，実空間コンテキストの処理負荷は膨大になるため，データ数に対する規模性が必要である．データ数の増加に対して，検索時間を平均的に少なくするにはインデクシングが効果的であることが知られている．インデクシングは条件比較対象となるデータ数を減らす事により検索効率を高める手法である．インデクシング手法は広く研究され，多くの分野で検索効率を高めるために用いられている．B-tree，B^+-tree は有名なインデクシング手法の一つである．これらの多くはデータの大小関係による二分木構造を持ち，一回の比較毎に検索対象数をおよそ半分に絞り込み $O(log(N))$ に近い検索時間を実現する．

しかし，領域データは多次元であるため大小関係が一意に定まらない．領域データを扱う実空間コンテキストシステムにおいて，検索効率を高めるインデクシング手法を考案することは研究課題の一つである．領域検索で最も有名なインデクシング手法の一つに R-tree がある．世界的に普及しているデータベースシステムである MySQL[10] と Postgresql[11] は領域検索に R-tree を用いている．R-tree はいくつかの領域データを包含するインデクスを階層的に構成し，検索時における条件比較対象のデータ数を絞り込むことで検索効率を高める（図 3.2）．図 3.2 は，階層構造によるインデクシング手法が，大小関係を持たない領域検索においても効果的である事を定量的に示している．従って，データ数に対して規模性を持つためには階層構造を持つ事が効果的である．

3.4 検索と比較して高負荷なデータ更新

実空間コンテキストの解析において，規模性を持つ情報処理性能を考慮する上での問題点は，実空間コンテキストが頻度高く更新する場合があるという特性に対応するのが困難なことである．R-tree は対象の情報が可変であり，多量になることが見込まれるシステムにおいて，情報検索能力の観点から適切であるとは言えない．この問題

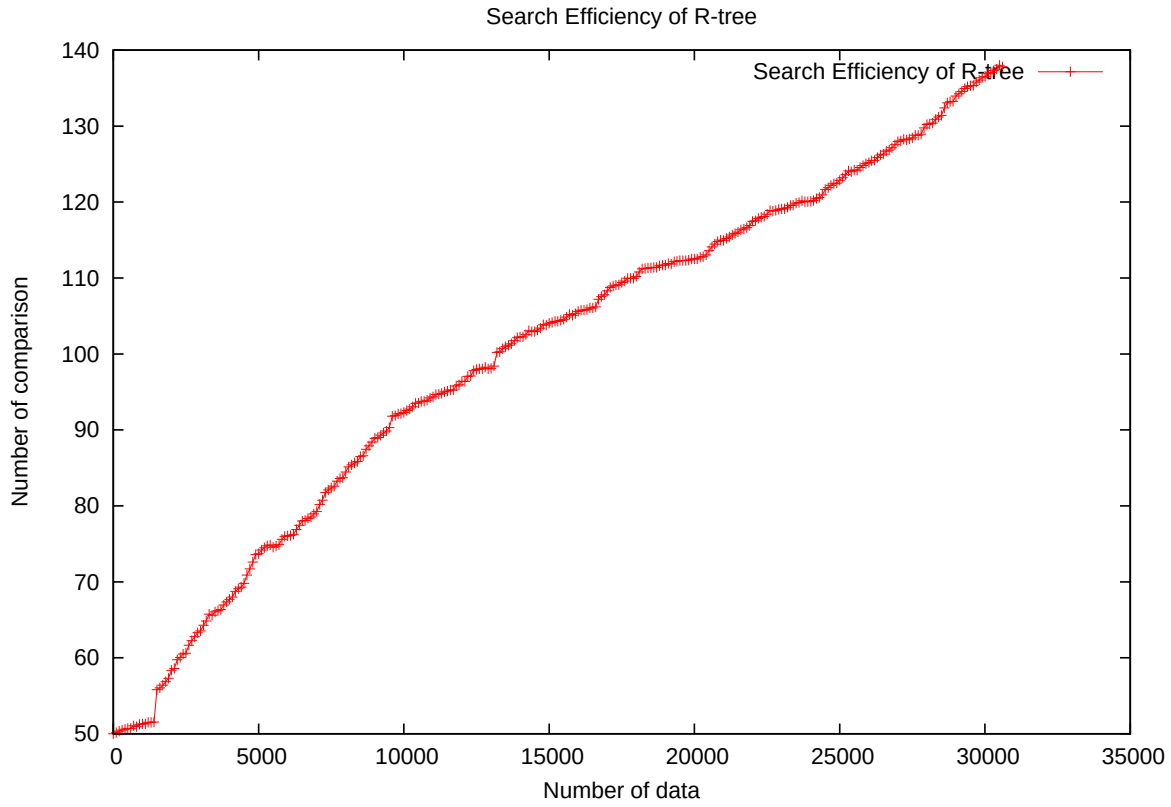


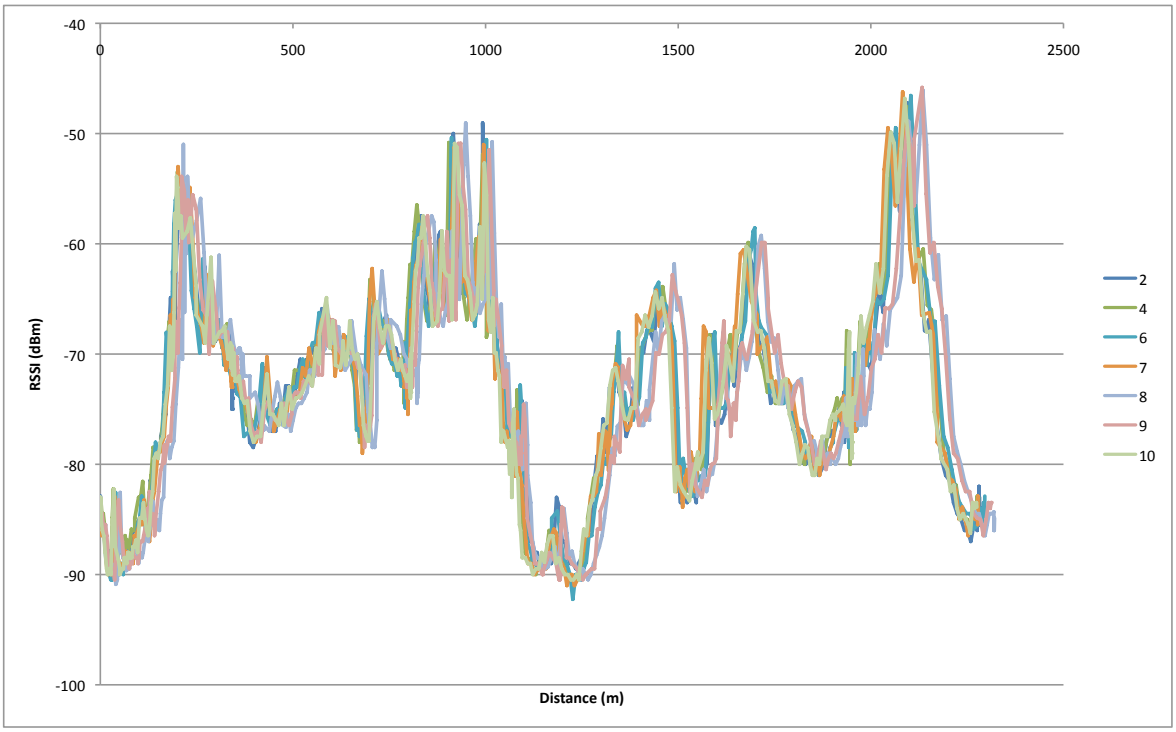
図 3.2: R-tree のノード数変化に対する検索時間（比較回数）

はデータの追加，削除，更新などの操作により，インデクスの再構成を必要とする範囲が広いためであり，複数の領域を包含する領域によりインデクスを構成する手法に起因する．実空間コンテキストにおける効率的な検索を行うためには，この問題を解決する必要がある．

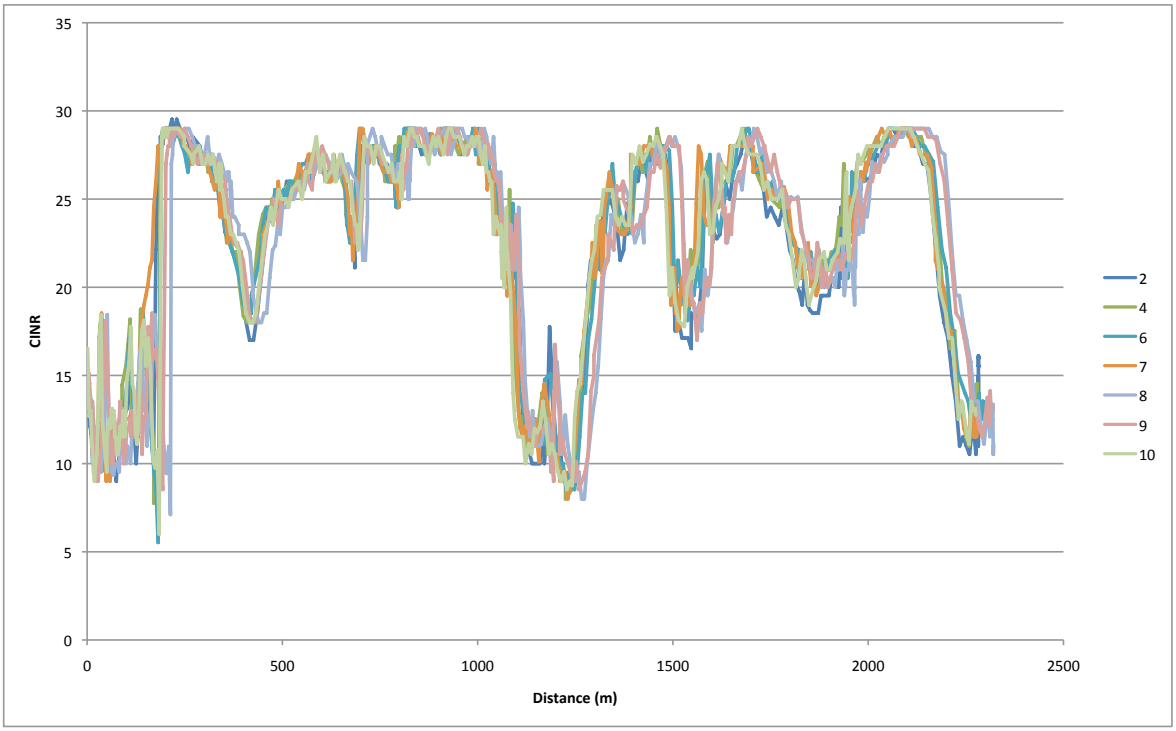
3.5 通信品質によるデータ取得時間の変動

コンテンツデータを取得する時間は，データサイズと通信実効速度により決まる．無線通信における通信実効速度は通信環境により異なる（図 3.3）．そのため，コンテンツデータを取得する時間は不定であり，有線通信に対して比較的大きく変動する．想定した時刻にコンテンツを表示することが求められる状況において，場合により表示が遅れることは問題である．広告配信では表示時間に応じて課金することが想定される．広告の表示時間を想定した時間に合わせるには，表示開始時刻と終了時刻を想定通りに合わせなくてはならない．通信環境の変動によりコンテンツ表示までの所要時間が揺らぐのは，リアルタイムに Web を表示する機能における問題である．

走行路（ルート）が異なると通信品質が変動するため，コンテンツを取得してから表示するまでの時間は変動する．第 2 章で述べた地域 WiMAX サービス環境を利用した



(a) RSSI



(b) CINR

図 3.3: 走行路における通信品質の変動

移動デジタルサイネージシステム（図 2.2）において，Web コンテンツの取得を開始した時刻から，コンテンツ表示を開始する時刻までの所要時間の変動を調べた結果を図 3.4 に示す．図 3.4 は比較的通信品質のよいルート A と，比較的通信品質の悪いルート B の二つの経路において，複数のコンテンツ（それぞれのサイズはルート A ではおよそ 30KB，34KB，52KB の三種類であり，ルート B ではおよそ 3KB，5KB，7KB，9KB，20KB，34KB，53KB，523KB，631KB の九種類）を周期的に取得した場合の，コンテンツ表示までの所要時間の確率分布とその累積確率を示している．ルート A では比較的安定してコンテンツを取得でき，分布が短い時間に集まっているが，ルート B ではコンテンツの取得時間の分布が広くばらついていた．無線通信回線品質が良くない場所では，コンテンツを取得する時間がばらつくことが定量的に示されている．

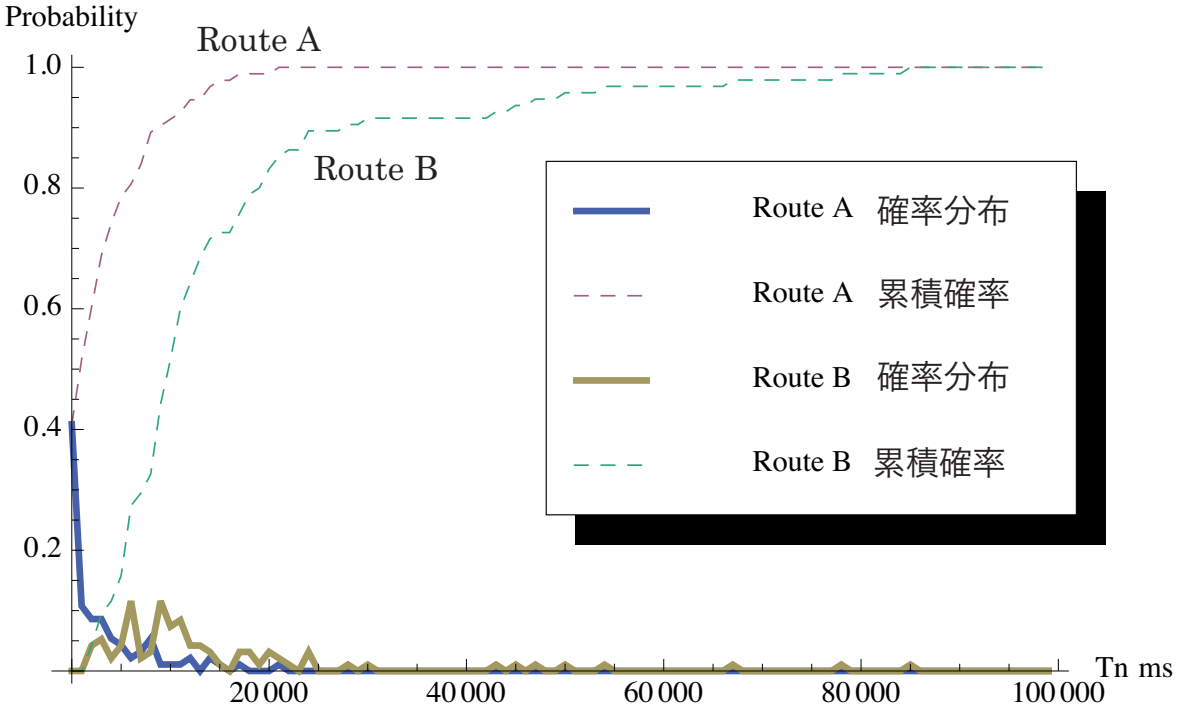


図 3.4: 異なる通信環境によるコンテンツ表示までの所要時間の変動

コンテンツデータを取得するために必要な時間は，通信実効速度からおよその推測が可能である．RTT(Round Trip Time) は IP 通信上のデータが機器間を往復するまでの所要時間であり，インターネット通信の実効速度を測定する指標として利用できる．一方，無線通信における電波状態の品質を表す指標に CINR(Carrier to Interference and Noise Ratio) がある．CINR により RTT が十分に推測可能であれば，通信にかかる時間を算出してコンテンツを表示するまでの時間が推測できると考えられる．

ここで，RTT により CINR が推測可能であるかを調べた．先に述べた WiMAX サービス環境において，RTT を Ping コマンドを用いて計測し，CINR を Netimizer[12] を用いて計測した結果を図 3.5 に示す．図 3.5 は CINR と RTT の散布図である．任意の時刻における CINR と RTT の値の組みを，CINR を横軸に RTT を縦軸にとってプロッ

トした．結果，相関係数は −0.34 であり，CINR と RTT との間に明確な正の相関関係は期待できないと分かった．

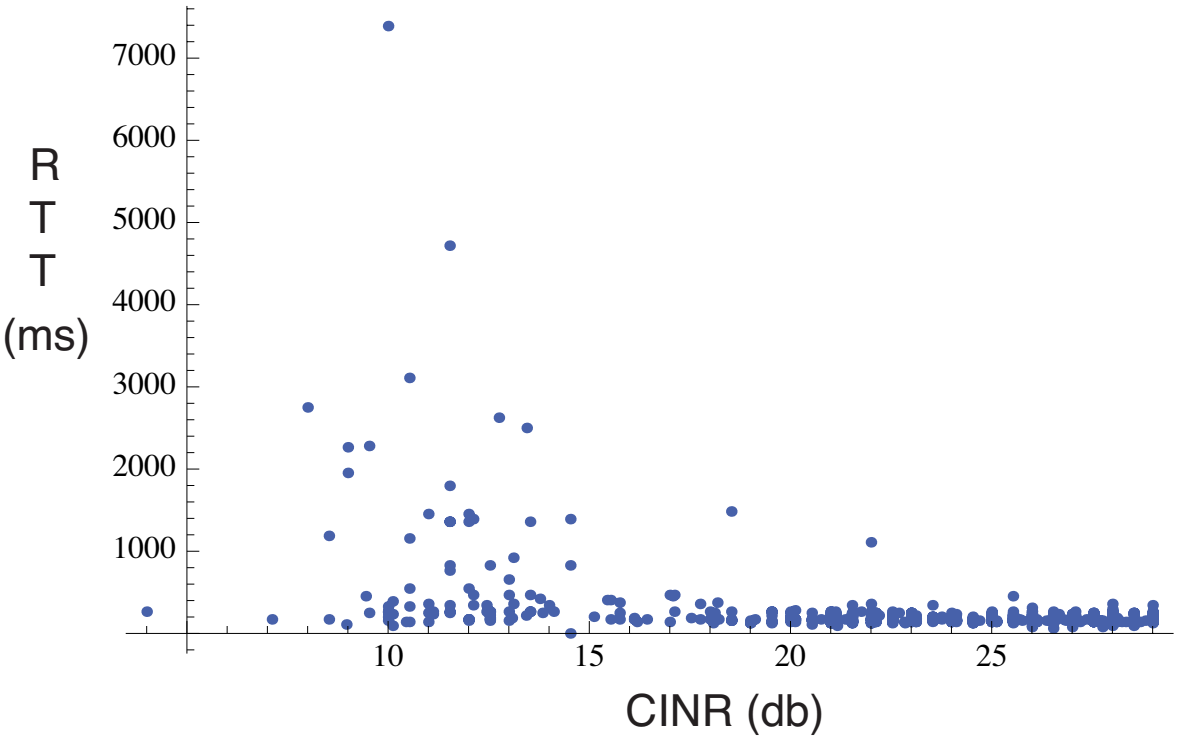


図 3.5: CINR と RTT の相関

コンテンツを表示するまでの所要時間は通信環境に応じて変動する．そのため，通信環境が有線通信に対して比較的大きく変動する移動体の無線通信環境においては，コンテンツを安定的に表示できない問題がある．通信実効速度の測定によりコンテンツデータ取得時間を推測し，コンテンツの表示開始時刻を調節することも可能であるが，無線通信環境と実効速度との強い正の相関を期待できず難しい．

3.6 まとめ

実空間コンテキストに応じた情報配信システムに対する要件を整理した．筆者らが構築した情報配信システムである，移動デジタルサイネージシステムの構築において得られた知見に加え，将来的な展望を考慮する事により，より一般的なシステムとして，実空間コンテキストに応じた情報配信システムへの要件を示した．結果，規模性と即時性の特性を同時に実現する必要があることを示した．

規模性の実現には，複数の実空間コンテキスト条件を高速に検索するインデクシング手法が必要である．即時性の実現には，コンテンツの取得時間を高確率で予測可能にし，確実にデータを取得する必要がある．

第4章　実空間コンテキストに応じたコンテンツ配信の関連研究

実空間コンテキストに応じたコンテンツ配信に関連した既存研究を示す．既存研究は空間データの検索を効率化する手法と，コンテンツを安定的に配信する手法について述べる．

実空間コンテキストの比較には空間データの検索が必要である．例えば，移動体の現在位置に応じてコンテンツを配信するためには，配信する対象の範囲に対して，移動端末の現在位置が含まれているかを比較する必要がある．移動体の現在位置は空間データであるとともに物理コンテキストである．このような空間データの検索は，検索対象のデータのインデクスを構成（インデクシング）する事で効率化される．即時性を実現するには空間データの検索を効率化する必要がある．空間データをインデクシングする手法は，主に，多次元の空間データをインデクシングする手法と，高次元のデータを低次元にマッピングする手法に分けられる．本章では双方の手法に対する既存研究について示す．

4.1　多次元の空間データに対するインデクシング手法

Guttman[13] は 2 次元の空間データに対するインデクスを構成するための汎用的な方法として，複数の空間データを包含する空間データによる木構造を持つ R-tree（図 4.1）を提案している．R-tree は対象のデータ群から Minimum Bounding Rectangle (MBR) となる矩形データを構築し，構築した矩形データ群に対してインデクシングする．例えば R-tree は，点のデータを 2 次元における単位矩形とし，直径が n である円のデータを，長さが n の正方形とし，単位矩形と長さが n の正方形を共に包含する MBR を構築する．対象のデータから構築した矩形データを R_0 とし，R_0 を包含する矩形データを R_1 すると，R-tree は対象データの追加に応じて対応する R_0 を構築し，R_1 を R_0 の MBR となるよう更新する．R_0 が一定数（m）を超えて N になったとき，R_1 が $\frac{N}{m}$ 個となるように分割する．R_0 に対する処理と同様に，分割された R_1 を包含する矩形データ（R_2）を構築する．同様に R_{n-1}(n は $N > \frac{N}{m^n}$ を満たす整数の最大値) に対してインデクス R_n を構築する．

Beckmann[14] は R-tree におけるインデクスの再構成を効率化した木構造のインデクス（R^*-tree）の構成を提案している．R-tree や R-tree から派生したインデクスの構成手法は高い頻度で変化する情報を効率的に扱えない．この問題は，情報が更新される

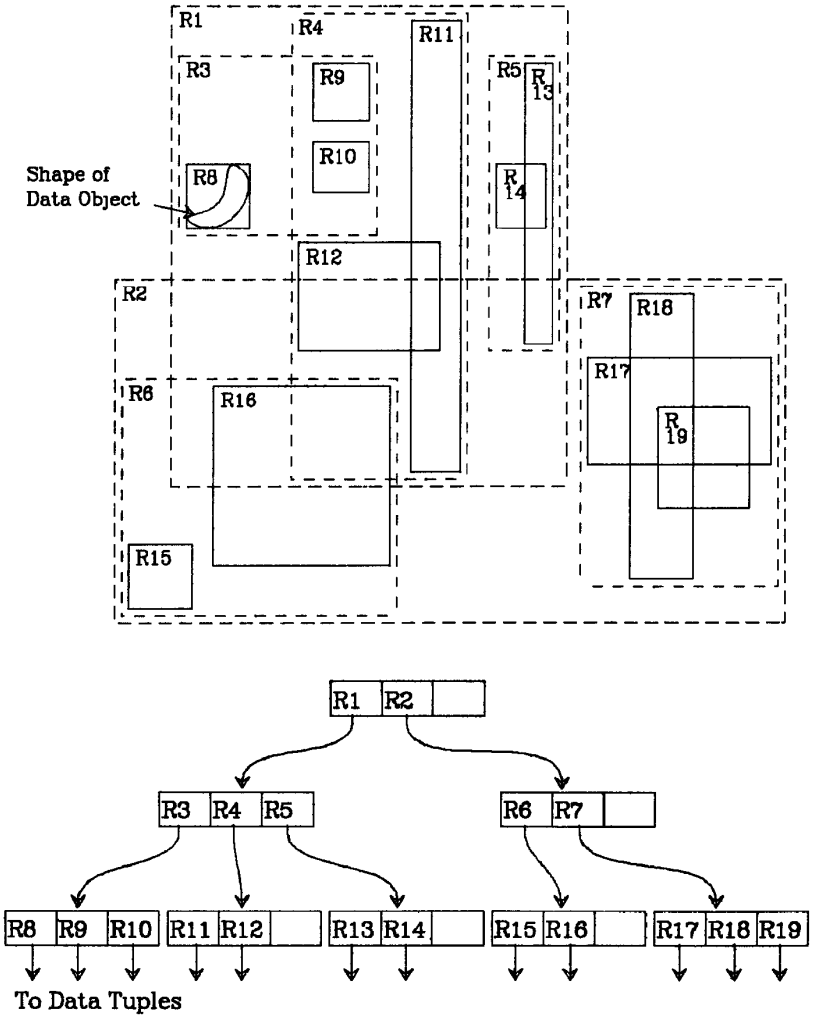


図 4.1: R-tree によるインデクシング例

によりインデクス情報を再構成する必要があることに起因する．空間データの検索を効率的に行うためには，インデクスの再構成を高速で行うことが課題である．Lee[15] はこの問題に対し，R-tree のインデクスを再構成するために 1 次元の情報に対するハッシュテーブルを用いて再構成の高速化を図っている．

4.2 多次元の空間データを 1 次元化するマッピング手法

空間充填曲線を用いると，多次元のデータをより小さい次元のデータで表せる．Finkel[16] は，空間充填曲線の再帰構造を用いて正規化された階層構造を持つ領域による Quad-tree（図 4.2）を提案している．例えば 2 次元の領域のデータは，1 次元での空間充填曲線上の点のデータで表せる．しかし，空間充填曲線は次元を減らす際に正規化されるため，任意の領域のデータを表すためには複数の領域を組み合わせる必要がある．そ

のため，インデクスを更新するには複数のインデクスをすべて更新する必要があり，効率的に更新を行う事ができない．

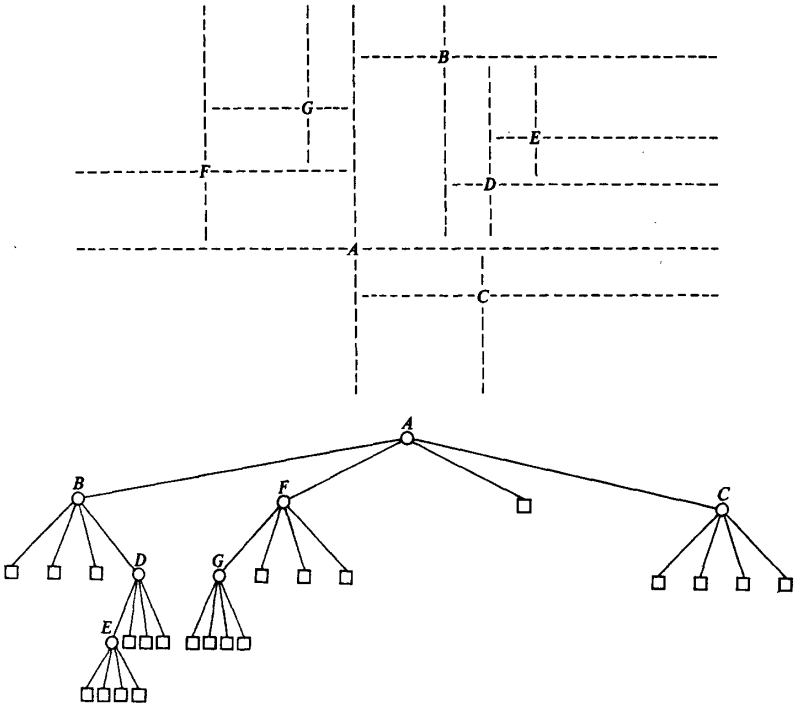


図 4.2: Quad-tree によるインデクス構成例

4.3 移動体に対するコンテンツデータの配信

移動体に対するコンテンツデータの配信は，コンテンツデリバリネットワーク（CDN）の分野で広く研究されてきた [17, 18, 19]．Yoshimura[19] らは，これらの研究の多くは，移動環境におけるリアルタイム性が必要とされるコンテンツとして動画配信を取り上げている．従来研究では，ネットワークベースのプリフェッチや，ネットワークにおけるキャッシュの配置，移動体の位置情報により，移動した先でコンテンツを効率よく取得できるネットワークシステムを提案している．

4.4 研究課題

R-tree や Quad-tree は階層構造を持つインデクスを構成して検索の高速化を実現している．しかし，時々刻々と変化するデータを多量に処理することに優れていない．これは，インデクスを再構成するために必要な処理が検索処理に対して比較的大きいためである．実空間コンテキストを扱うためには，移動する物体の位置を対象とする物

体の数だけ扱う必要がある．そのため，既存の手法では即時性を持つデータを効率的に検索することができない．

既存の研究の多くは多次元の情報を一つのツリーで同時に扱える．位置の情報は 1 次元のデータで表し，範囲を持つ領域の情報は 2 次元のデータで表すと，双方に比較可能な形式であるため同時に扱える．同時に扱える情報は同一空間の情報に限る．位置と範囲を持つ領域の情報は，同じ物理空間の情報であるため，同一空間の情報である．しかし，物理コンテキストは位置や範囲を持つ領域の情報に限らず，温度や湿度などのセンシング情報も含む．実空間コンテキストは多様なデータを含み，その一つである物理コンテキストに対して，既存の手法を直接用いて効率的に検索することはできない．

実空間コンテキストを効率的に検索するためには，時々刻々と変化するデータを多量に処理できる即時性に加え，多量のデータを高速に処理するだけの規模性を持つ検索手法の検討が研究課題となっている．

第5章　アプローチ

実空間コンテキストに基づいたコンテンツ配信を行うためには，二つの特徴を持つシステムが必要であることを第 3 章で述べた．一つは，**条件コンテキストと実空間コンテキストとの検索を効率的に行う手法**である．もう一つは，比較の結果より求まる配信する必要のあるコンテンツを，配信先端末の移動を考慮しつつ，**安定的かつできる限り早く配信する手法**である．

5.1　条件の関係に基づく検索手法の提案

第 3.4 節における課題を踏まえ，複数の条件コンテキストと実空間コンテキストとを効率的に比較する手法を提案する．ここで，複数の条件のうち，必要または十分などの関係性に着目し，十分条件を予め比較することにより，いくつかの条件を比較する必要を省いて効率化する手法を考える．

5.1.1　条件の関係と効率化

いま，任意の条件 A, B を考える．条件 A, B の関係は表 5.5 に示すように，次式 5.1 の 4 つの場合に分類できる．

$$\begin{cases} A \cap B = \emptyset \\ A \cap B \neq \emptyset \\ A \cap \overline{B} = \emptyset \\ \overline{A} \cap B = \emptyset \end{cases} \tag{5.1}$$

A	B
1	0
0	1
0	0

表 5.1: $A \cap B = \emptyset$ $(A \Rightarrow \overline{B})$

A	B
1	1
1	0
0	1
0	0

表 5.2: $A \cap B \neq \emptyset$

A	B
1	1
0	1
0	0

表 5.3: $A \cap \overline{B} = \emptyset$ $(A \Rightarrow B)$

A	B
1	1
1	0
0	0

表 5.4: $\overline{A} \cap B = \emptyset$ $(B \Rightarrow A)$

表 5.5: 条件 A,B の関係ごとの真理値表

ここで，次式5.2に示す通り，条件が交差する場合を除く3つの場合において，一方の条件の真偽より他方の条件の真偽が分かる場合がある．

$$\begin{cases} A \cap B = \emptyset, A \Rightarrow \overline{B} \\ A \cap \overline{B} = \emptyset, A \Rightarrow B \\ \overline{A} \cap B = \emptyset, B \Rightarrow A \end{cases} \tag{5.2}$$

従って，条件同士の関係により，任意の条件集合において，条件比較回数を条件集合の数より少なくしつつ，全体の条件を調べることが可能である．特に，実空間コンテキストを扱うアプリケーションにおいて，条件が包含関係にある場合は多く存在すると考えられる．例えば，東京都を配信対象としつつ，東京の区毎に配信するコンテンツを変え，更に各区のランドマーク周辺で配信するコンテンツを指定する場合は，配信する場所が条件であり，条件同士の関係が包含関係となる．よって，条件の関係に基づく検索を行うことで，実空間コンテキストの比較を効率化できると期待できる．

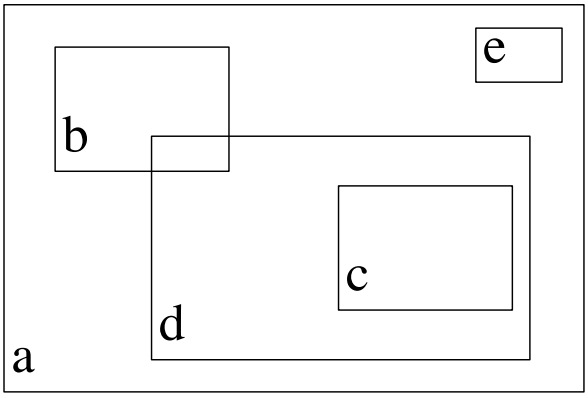


図 5.1: 式5.3における条件の関係

条件同士の関係性を求めるには，各条件同士の比較を行えばよい．このとき，存在しうる関係の数は条件の数 N に対して 2^N である．5.1に示した通り，関係の組みによっては，条件の真偽の組み合わせが存在しない場合があるため，存在する条件の組み合わせは 2^N より少ない．ここで，存在する条件の組み合わせより，効率的に比較操作が可能な，条件の組み合わせの順序を発見することが問題である．いま，条件集合 C において，C の要素 $a, b, c, d, e(\in C)$ を考える．ここで，a, b, c, d, e は図5.1に表されるように，次式5.3の関係を持つとする．

$$\begin{cases} b \vee c \vee d \vee e \Rightarrow a \\ \neg(\exists b \vee d) \\ c \Rightarrow d \\ e \Rightarrow \overline{b} \vee \overline{d} \end{cases} \tag{5.3}$$

すると，a, b, c, d, e の起こりうる組み合わせをは7通り存在する（表5.6）．これより，a, b, c, d, e の効率のよい比較順序の発見は表5.6（左側）の数値を整頓する問題と考え

a	b	c	d	e	
0	0	0	0	0	(行 1)
1	0	0	0	0	(行 2)
1	0	0	0	1	(行 3)
1	1	0	1	0	(行 4)
1	1	0	0	0	(行 5)
1	0	1	1	0	(行 6)
1	0	0	1	0	(行 7)

a	b	c	e	d	
1	0	1	0	1	(行 6)
1	1	0	0	1	(行 4)
1	0	0	0	1	(行 7)
1	0	0	1	0	(行 3)
1	1	0	0	0	(行 5)
1	0	0	0	0	(行 2)
0	0	0	0	0	(行 1)

表 5.6: 関係式 5.3 における条件の真理値表（左側：整頓前，右側：整頓後）

られる．整頓した結果は表 5.6（右側）となる．結果，d, e, c, b, a の順で比較すると，平均 4.2 回の比較で組み合わせを調べることができ，効率化できることが分かる．

5.1.2 効率的な条件比較順序の導出

条件の関係を求め，求めた関係を整頓することにより，条件集合から効率的に比較できる条件の順序を導く．条件関係が整頓されている状態とは，条件集合の真理値表において，一番右の列の上から下に真理値を見たときに，1 が連続した後 0 が一続きになっている．また，一つ左の列を 1 が連続している行と 0 が連続している行とを分けて，列を上から下に真理値表を見たときに，同様に 1 が連続した後に 0 が一続きになっている．このように真理値表が整頓されているとき，条件集合から任意の条件に一致する条件の組み合わせを比較するには，真理値表の一番右の列の条件から比較を始め，比較結果 (0 か 1）に応じて調べる対象の行を絞り，1 行に絞り込むまで繰り返せば良い．比較の結果残った行が，任意の条件と一致する条件の組み合わせである．

次に，条件集合の真理値表を整頓する方法について述べる．整頓された真理値表における一番右の列から見た条件の並び順を，整頓された条件順序と呼ぶ．まず，条件集合がただ 1 つの条件 a を含む場合，a の真理値表は，1 と 0 が一つだけある表であり，この状態は整頓されている．ここで，整頓された状態の条件集合の真理値表を基準として，任意の条件 a_m を追加する場合を考える．いま，条件集合を $C(C \ni a_n\{n|0..N\})$ とし，C において存在する値の組み合わせを P とする．真理値表において，C は一番上の行の条件集合を示し，P はそれ以外の行を示す．P は条件集合における全集合であるため，P の各要素と a_m との関係を調べることで，a_m を追加した後の P を導出できる．P の各要素との比較は，整頓された条件順序の順に比較することと等しい．従って，整頓された条件順序に従って a_n と a_m を比較し，重なった条件が存在する場合における a_n の前に a_m を挿入すれば，整頓された条件を保つことができる．

5.2 領域の包含関係に基づく検索ツリー (AR-tree) の提案

条件の関係に基づく検索を実現する検索ツリー **Area Relation Tree (AR-tree)** を提案する．AR-tree は第 5.1 節で述べた，整頓された真理値表と同等の構造を持つ．条件は任意の空間を，条件を満たす部分と満たさない部分に分ける．これは，地理データにおける領域が中と外に分けるという関係と同様である．条件を拡張して地理データとして考えることができ，AR-tree により地理データの関係に基づいた検索を行うことができる．以下，AR-tree による地理データの検索について述べる．

AR-tree は領域の包含関係による二分木構造を持つインデクスツリーである．AR-tree を用いることで任意の点を含む領域が検索できる．AR-tree はインデクスノードとデータノードより構成される．インデクスノードは領域データに加え，領域データの包含関係によるインデクスノードとの接続関係を持つ．包含関係は領域を含む場合と含まない場合の二通りである．ここに a から e の領域データがある場合，図 5.2 の構造を持つ AR-tree を構成できる．a に着目すると，e は a に含まれ，a に含まれない領域は存在しないため e が唯一の子である．また，b に着目すると，b と d は共有する領域を持つため，b は d と共有する領域による親子関係と，d と共有しない領域による親子関係の二通りの関係を持つ．任意の領域同士の関係は，含むか，重なるか，含まないかのいずれかに決定できるため，重なる場合における領域を，含む場合と含まない場合の領域に分ける事で，AR-tree は任意の領域データに対して構成可能である．

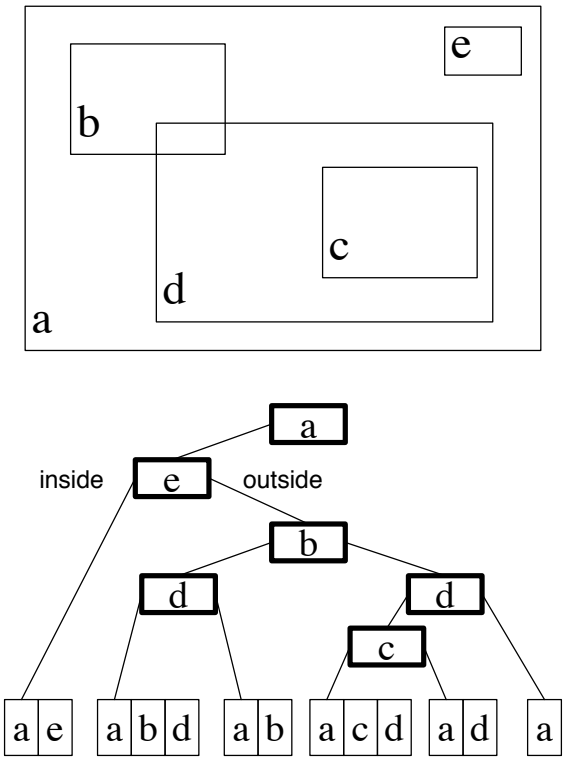


図 5.2: AR-tree によるインデクシング例

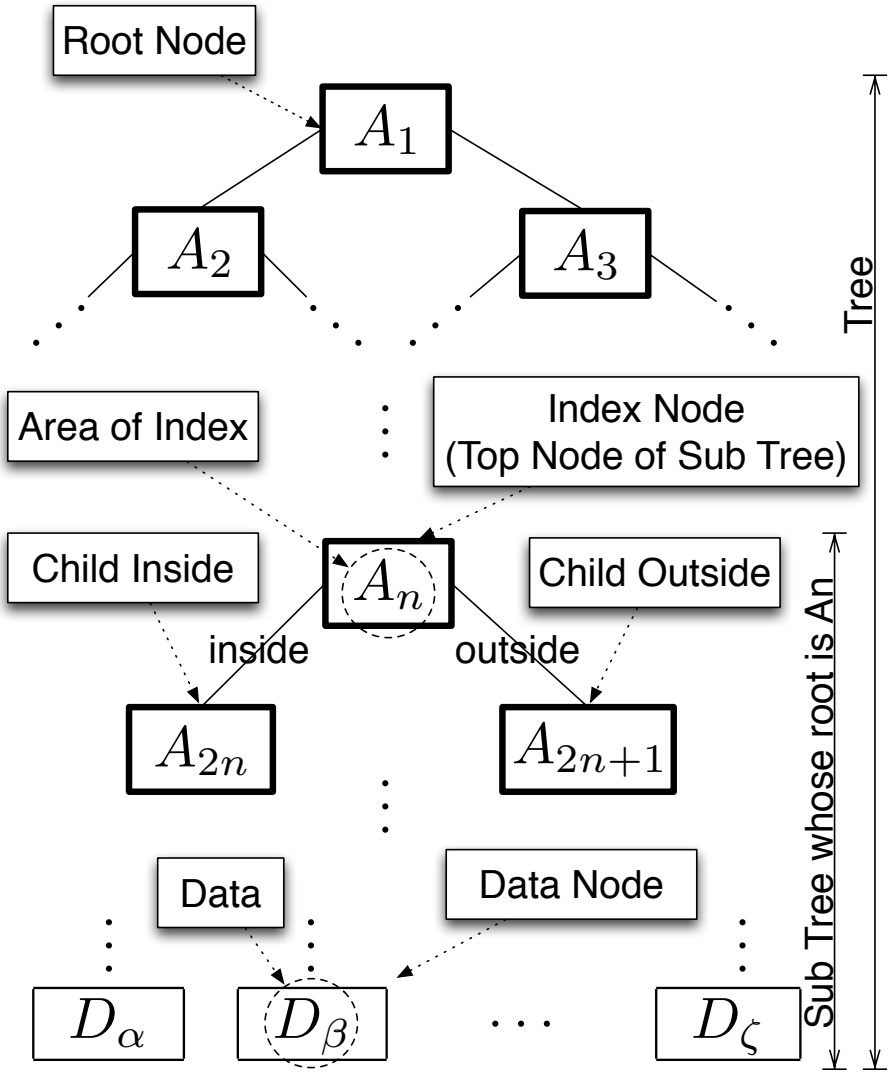


図 5.3: AR-tree の構造と各部名称

5.2.1 AR-tree の特徴

AR-tree は領域データの集合をインプットとし，インプットデータの冪集合を Data Node が持つデータとする．Data Node が持つデータ集合はツリーにおける，Root から任意の Data Node までのパスと対応する．Data Node が持つデータ集合とツリーのパスとの関係性を図 5.4 に示す．図 5.4 は，Index Node を横軸に並べ，Index Node 間の包含関係（Inside または Outside）を縦軸にとった図である．図中の実線は Root から Data Node までのパスを示す．ここで，Data Node D_α が持つデータは，縦軸が Inside であるときの横軸が示す領域の集合と関連づけられたデータ集合である．

AR-tree は階層構造を持つ．階層構造とは，任意の節を最上位とするサブツリーが，Root を最上位とするツリーと同じ特徴を再帰的に持つことである．ここで，図 5.2 に

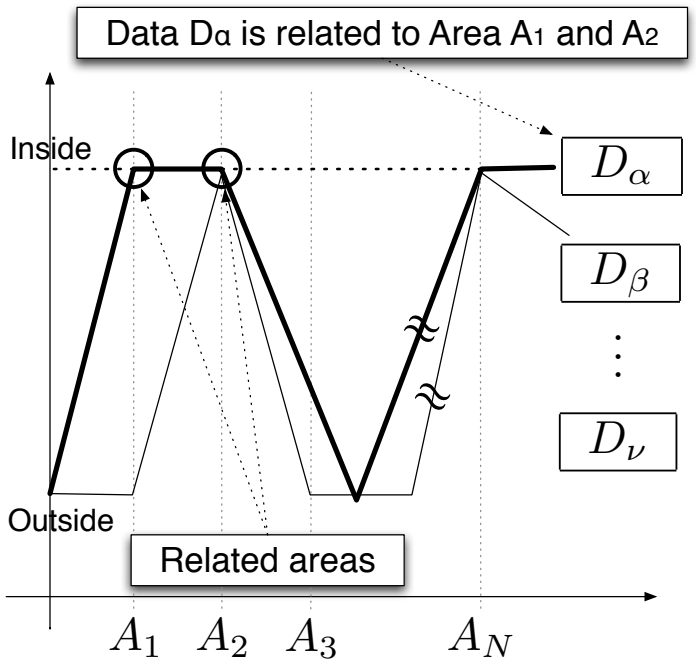


図 5.4: AR-tree のデータと領域の関係

おけるインデクスの階層構造に着目して図 5.5 に示す．図 5.5 より AR-tree が階層構造を持つことが分かる．

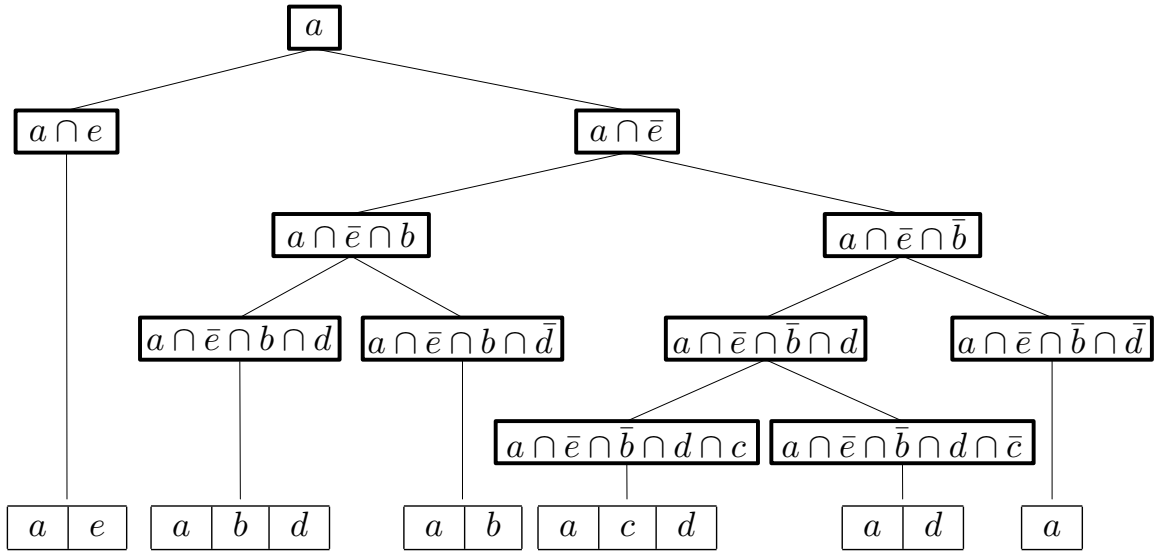


図 5.5: AR-tree の階層構造を示したインデクシング例

AR-tree は，任意の点を含む領域の検索結果と，ただ一つの Data Node が持つデータが対応する．この特徴はツリーの Root から点の検索を行った結果，比較したインデ

クスの軌跡がただ一つの線になることを意味する．従って，任意の点を含む領域を検索する過程において，ツリーの分岐検索がおこらない．

5.2.2 点の検索アルゴリズム

点（KeyPoint）を検索するアルゴリズムを図 5.6 に示すとともに以下に述べる．領

```
SearchPoint(AR-tree KEY_POINT)
1  if AR-tree is empty  then
2    Point KEY_POINT is not included any areas.
3  else
4    do while TOP_NODE of tree is index
5      if area of TOP_NODE contains KEY_POINT  then
6        Continue loop on subtree whose top is inside child of TOP_NODE.
7      else if area of TOP_NODE and KEY_POINT are disjoint  then
8        Continue loop on subtree whose top is outside child of TOP_NODE.
9      end if
10   end while
11   Point is included areas which have DATA_NODE.
12 end if
```

図 5.6: 点の検索アルゴリズム

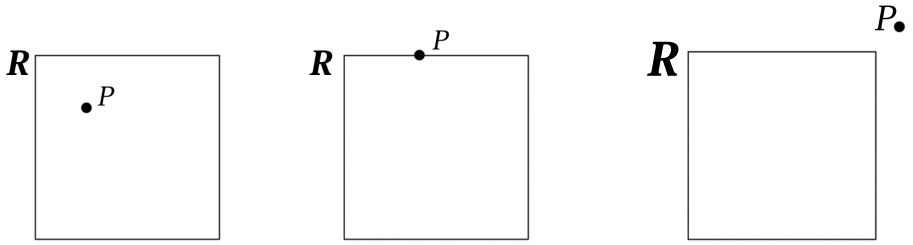
域の検索はツリーの Root が示す領域と KeyPoint を比較し始め，データを発見するまで KeyPoint と節が示す領域と比較を繰り返す．データの発見は，比較対象の領域と点の包含関係（図 5.7）より二つある子のいずれかを選択し，選択した子を最上位とするサブツリーに対して点の検索を再帰的に試みる．子の選択方法は比較される領域の包含関係より次の通り決定する．ただし，比較対象のうち，インデクスが示す領域を IndexArea とし，インデクスが持つ二つの子のうち，IndexArea に含まれる領域を持つ子 IndexNode を ChildInside，他方を ChildOutside とする．

- IndexArea が KeyPoint を含む場合：ChildInside を選択する
- IndexArea が KeyPoint に含まない場合：ChildOutside を選択する

検索のために必要な IndexArea と KeyPoint の比較回数 $S(compare)$ はツリーの深さ $d\{d_{D_\alpha}, d_{D_\beta}, \ldots, d_{D_\zeta}\}$ を用いて，

$$MIN(d) < S(compare) < MAX(d)$$

と表せる．



(a) contain ($p \in R$)　　(b) disjoint ($p \notin R$)

図 5.7: 領域 R から点 P への関係

5.2.3 領域の追加アルゴリズム

領域（NewArea）を AR-tree に追加するアルゴリズムを図 5.8 に示すとともに以下に述べる．ここで，領域の追加アルゴリズムをツリーまたは任意のサブツリーと NewArea を用いて，AppendArea((Sub)Tree, NewArea) とおく．

領域の追加は，ツリーの各データに関連する領域群と NewArea により，新たな領域群を作ると同時に対応する検索ツリーを作る操作である．ここで，NewArea を N，追加する対象の AR-tree における任意の IndexNode を A_n，A_n の親から Root までの IndexNode が持つ領域を結合した領域を P_n．A_{2n} を最上位とするサブツリーを I_n．A_{2n+1} を最上位とするサブツリーを O_n とする．

いま，領域 N を追加する問題は，領域 P_n に対し，$P_n \cap N$ と $P_n \cap A_n$ の 2 つの領域による新たな領域作成の問題と考えられる．従って，追加領域に対するインデクスを構築する方法は，2 つの領域 $P_n \cap N$ と $P_n \cap A_n$ の関係より考えればよい（図 5.9）．図 5.10 に示す通り，2 つの領域の関係は 4 つの場合が考えられる．$P_n \cap N$ と $P_n \cap A_n$ の関係による領域の追加方法を，4 つのそれぞれの場合において以下に示す．

場合 1 $P_n \cap A_n$ と $P_n \cap N$ が重なる場合 $((P_n \cap A_n) \cap (P_n \cap N) \neq \emptyset)$

パターン 1 N を A_n の子 IndexNode として追加する方法

1. I_n と O_n に対して N を追加

パターン 2 N を A_n の親 IndexNode として追加する方法

1. N を A_n の場所に挿入
2. A_n を N の Child Inside とし，A'_n を複製し，N の Child Outside とする
3. I_n をサブツリー $I'_n (I'_n \subseteq P_n \cap N)$ とサブツリー $I''_n (I''_n \subseteq \overline{P_n \cap N})$ に分け，I''_n を A'_n の Child Inside とする
4. O_n をサブツリー $O'_n (O'_n \subseteq P_n \cap N)$ とサブツリー $O''_n (O''_n \subseteq \overline{P_n \cap N})$ に分け，O''_n を A'_n の Child Outside とする
5. I'_n と O'_n の全ての Data Node に D_N を追加

AppendArea(AR-tree NewArea)

```
1  Let TOP_NODE be root of AR-tree.
2  begin routine AppendAreaToSubTree(TOP_NODE, KEY_AREA)
3    do while TOP_NODE is index
4      if area of TOP_NODE intersects KEY_AREA then
5        Do AppendArea(child which is inside of TOP_NODE, KEY_AREA).
6        Do AppendArea(child which is outside of TOP_NODE, KEY_AREA).
7        Break out of routine.
8      else if area of TOP_NODE and KEY_AREA are disjoint then
9        Continue routine with let TOP_NODE be outside child of TOP_NODE.
10     else if area of TOP_NODE includes KEY_AREA then
11       Create index NEW_INDEX as root with area of NEW_AREA.
12       Place TOP_NODE as inside child of NEW_INDEX.
13       Split subtree O_n into O_n_1 which is inside of area P_n and N
14         and O_n_2 which is outside of area P_n and N.
15       Append NEW_AREA to all data nodes for I_n and O_n.
16     else if area of TOP_NODE contains KEY_AREA then
17       Continue routine with let TOP_NODE be outside child of TOP_NODE.
18     end if
19   end while
20 Create index INDEX as root with area of NEW_AREA.
21 Place NEW_AREA as child of INDEX.
```

図 5.8: 領域の追加アルゴリズム

場合 2 $P_n \cap A_n$ と $P_n \cap N$ が重ならない場合 $((P_n \cap A_n) \supseteq (P_n \cap N) = \emptyset)$

パターン 1 N を A_n の子 IndexNode として追加する方法

1. O_n に対して N を追加

パターン 2 N を A_n の親 IndexNode として追加する方法

1. N を A_n の場所に挿入
2. N の Child Inside を I_n とし，Child Outside を O_n とする
3. O_n をサブツリー $O'_n(O'_n \subseteq P_n \cap N)$ とサブツリー $O''_n(O''_n \subseteq \overline{P_n \cap N})$ に分け，O''_n を A'_n の Child Outside とする
4. O'_n の全ての Data Node に D_N を追加

場合 3 $P_n \cap A_n$ が $P_n \cap N$ を含む場合 $((P_n \cap A_n) \supseteq (P_n \cap N))$

パターン 1 N を A_n の親 IndexNode として追加する方法

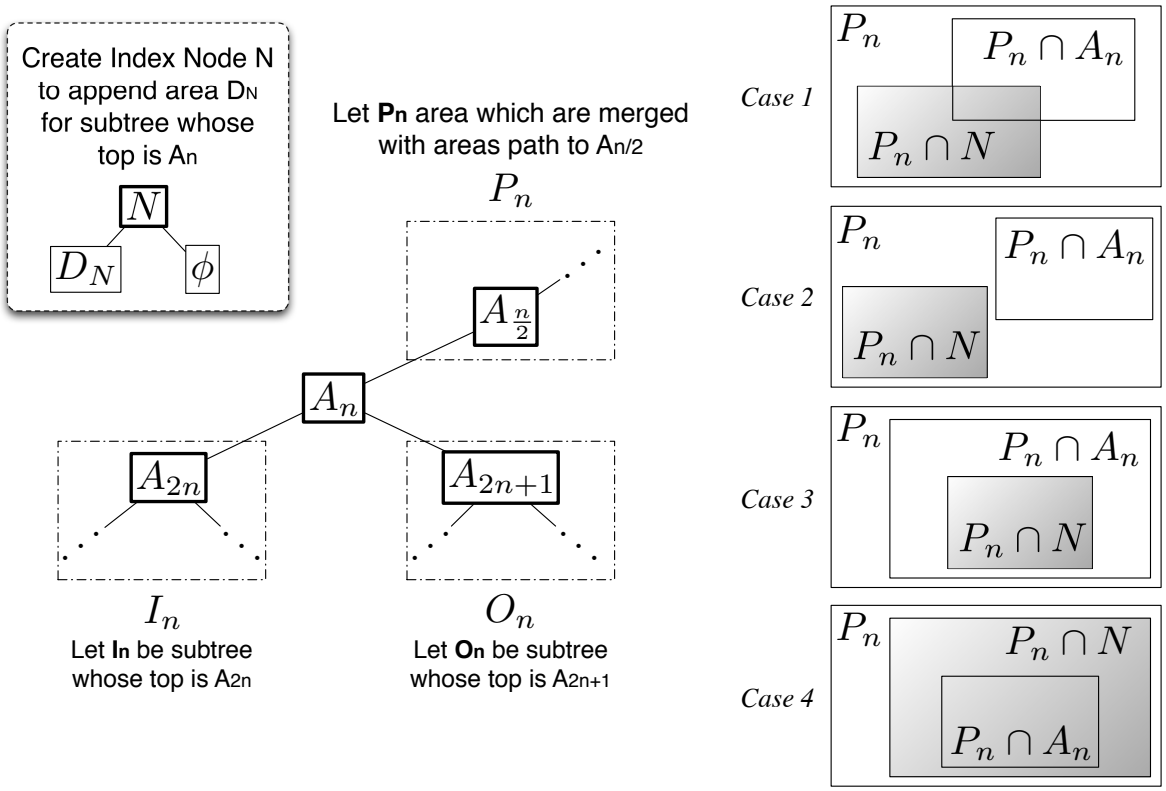


図 5.9: 任意の（A_n を最上位とする）サブツリーに対する領域 N の追加

1. N を A_n の場所に挿入
2. A_n を N の Child Outside とする
3. I_n をサブツリー $I'_n(I'_n \subseteq P_n \cap N)$ とサブツリー $I''_n(I''_n \subseteq \overline{P_n \cap N})$ に分け，I'_n を N の Child Inside とし，I''_n を A_n の Child Inside とする
4. I'_n の全 Data Node に D_N を追加

パターン 2 N を A_n の子 IndexNode として追加する方法

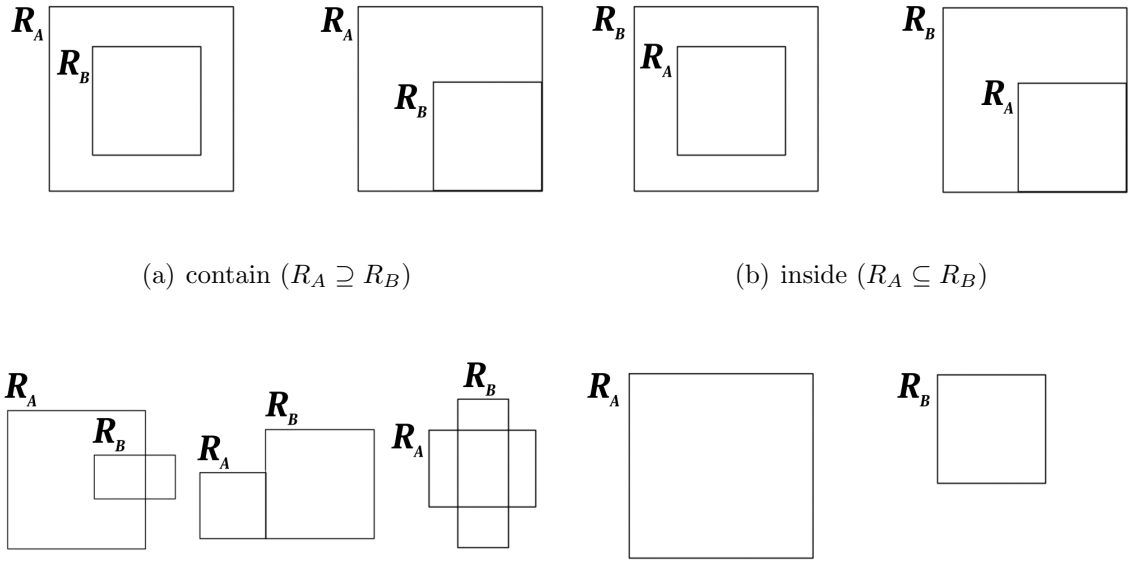
1. I_n に対して N を追加

場合 4 $P_n \cap A_n$ が $P_n \cap N$ に含まれる場合 $((P_n \cap A_n) \subseteq (P_n \cap N))$

パターン 1 N を A_n の親 IndexNode として追加する方法

1. N を A_n の場所に挿入
2. A_n を N の Child Inside とする
3. O_n をサブツリー $O'_n(O'_n \subseteq P_n \cap N)$ とサブツリー $O''_n(O''_n \subseteq \overline{P_n \cap N})$ に分け，O'_n を A_n の Child Outside とし，O''_n を N の Child Outside とする
4. I_n と O'_n の全ての Data Node に D_N を追加

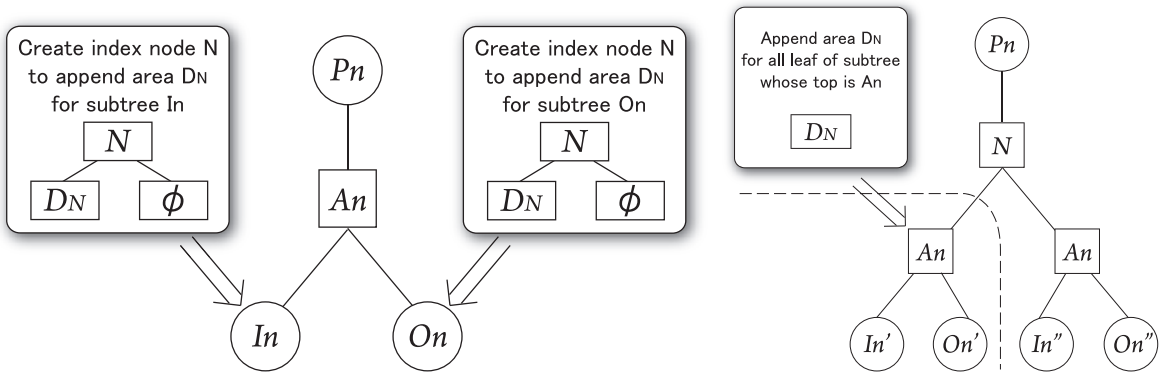
パターン 2 N を A_n の子 IndexNode として追加する方法



(a) contain ($R_A \supseteq R_B$)　(b) inside ($R_A \subseteq R_B$)

(c) overlap ($R_A \cap R_B \neq \emptyset$)　(d) disjoint ($R_A \cap R_B = \emptyset$)

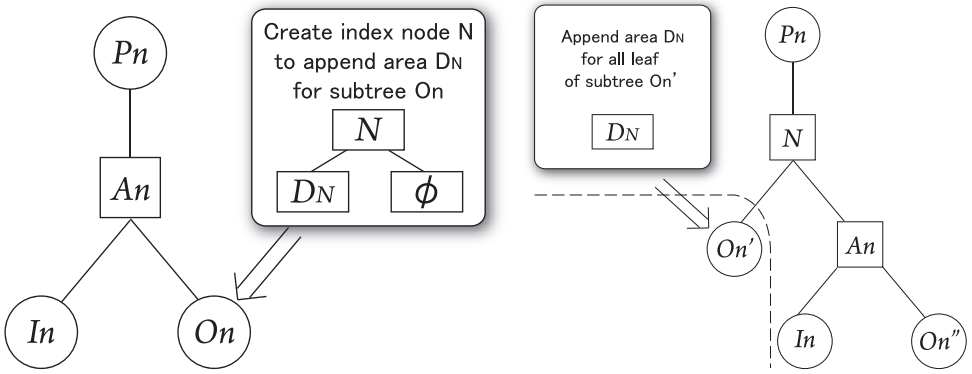
図 5.10: 領域 R_A から領域 R_B への関係



(a) pattern 1　(b) pattern 2

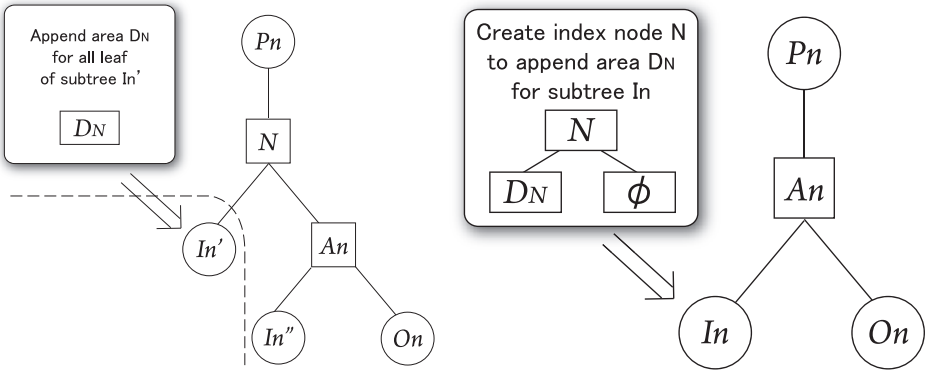
図 5.11: $P_n \cap A_n$ と $P_n \cap N$ が重なる場合（場合 1）の領域 N の追加方法

1. I_n の全ての Data Node に D_N を追加
2. O_n に対して N を追加



(a) pattern 1　　(b) pattern 2

図 5.12: $P_n \cap A_n$ と $P_n \cap N$ が重ならない場合（場合 2）の領域 N の追加方法



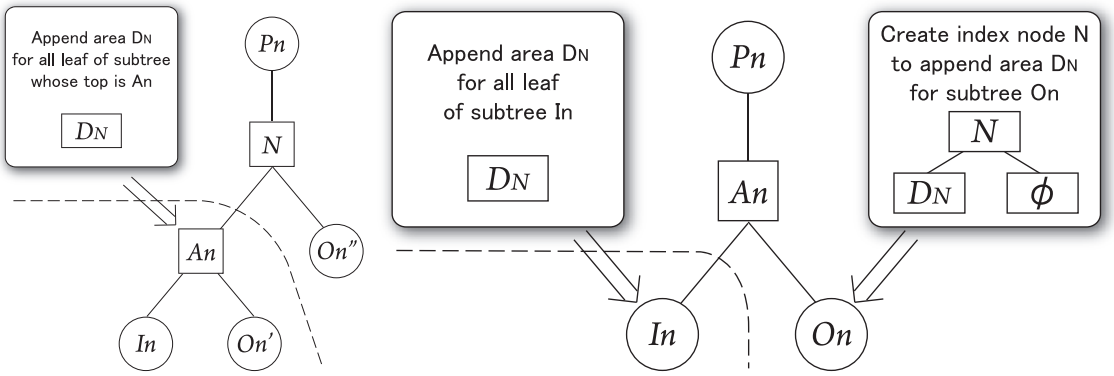
(a) pattern 1　　(b) pattern 2

図 5.13: $P_n \cap A_n$ が $P_n \cap N$ を含む場合（場合 3）の領域 N の追加方法

5.3　コンテンツの非同期プリフェッチの提案

第 3.5 節の問題を踏まえ，本研究ではコンテンツデータを取得する方式として，**非同期プリフェッチ**を提案する．非同期プリフェッチは，コンテンツの順序とは無関係に予め表示に必要なコンテンツを取得し，コンテンツデータの取得処理をコンテンツの表示処理とは非同期に行う方式である．

非同期プリフェッチの概要を図 5.15 を用いて説明する．既存のコンテンツデータ取得手法を**同期フェッチ**とし，非同期プリフェッチと比較する．同期フェッチは任意のコンテンツ n を表示する直前にコンテンツデータを取得する．コンテンツデータの取得を開始した時刻を $t1$ とし，表示を開始した時刻を $t2$ とすると，同期フェッチ，非同期プリフェッチともに，コンテンツ n を表示するまでの所要時間 T_n は $t2 - t1$ である．しかし，同期フェッチにおいてはコンテンツの表示中の時刻とコンテンツデータ取得中の時刻は重ならず，非同期プリフェッチにおいては重なることがある．さらに，任意の時



(a) pattern 1　　(b) pattern 2

図 5.14: $P_n \cap A_n$ が $P_n \cap N$ に含まれる場合（場合 4）の領域 N の追加方法

刻において表示順序とは無関係に取得できる．

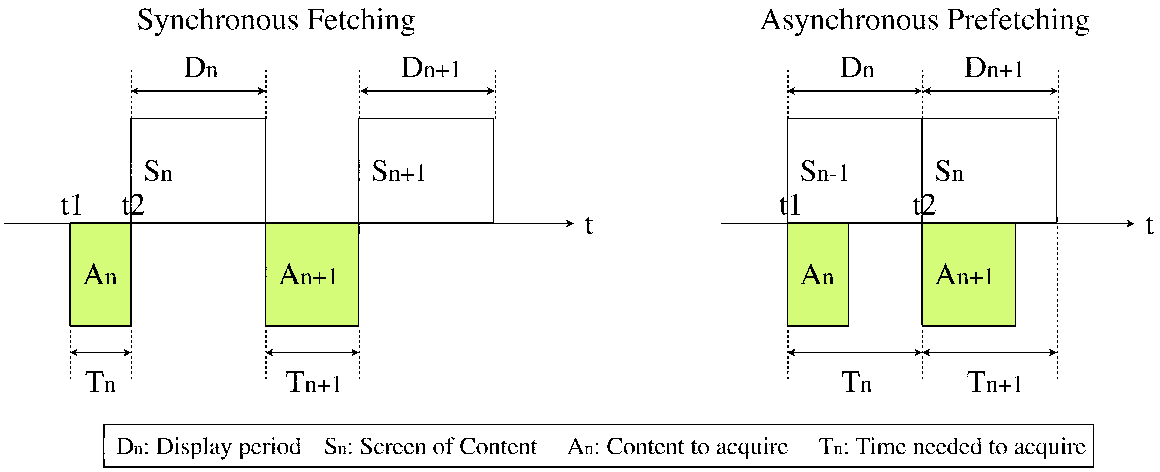


図 5.15: 同期フェッチと非同期プリフェッチによるコンテンツデータ表示までの所要時間

非同期プリフェッチにより，任意のコンテンツを遅れることなく表示することが可能である．これはコンテンツを取得する時間をもとにしたスケジューリングを行うことができるからである．非同期にフェッチを行うことで，コンテンツデータはコンテンツの順序とは無関係に取得できる．従ってコンテンツの表示順序に依らないコンテンツのプリフェッチが可能であり，表示を行う前にコンテンツを取得するための十分な時間を確保するなどの時間調節が可能である．事前にコンテンツデータの取得時刻を調節することにより，実効速度に応じた動的な時間調節も可能である．

5.4 まとめ

第3章で示した問題を解決するためのアプローチを示した．データ更新の負荷を低減するための手法として AR-tree を提案し，通信品質の変動を高確率で予測するための手法として非同期プリフェッチを提案した．AR-tree は，領域データに対して点の検索を行うためのインデクスツリーである．AR-tree は領域の包含関係による階層構造を持ち，インデクスのための領域を，領域データから作成しないため，データの更新に対するインデクスの再構成が不要である．非同期プリフェッチは，コンテンツを表示する処理と取得する処理を分離する手法である．データを非同期に取得する事により，コンテンツを表示するまでに，十分な時間を確保できるため，取得完了までにかかる時間が通信品質によらず予測可能である．

第6章　設計

第 5 章では，実空間コンテキスト条件と実空間コンテキストとの比較を効率的に行う AR-tree と，安定的かつできる限り早く配信する非同期プリフェッチの実現方法を述べた．本章では，実空間コンテキストに応じたコンテンツ配信を行うための，AR-tree と非同期プリフェッチを連携するシステムを設計する．2 つの手法を実現するシステムにより，実空間コンテキストに応じたコンテンツ配信を実現できる．

6.1　システム設計概要

図 6.1 は，実空間コンテキストに応じたコンテンツ配信システムの概要を示している．コンテンツの配信元はコンテンツ配信システムに指定したコンテンツの ID（URL）に加え，配信したい領域など実空間コンテキスト条件を登録する．一方，センサはセンシングデータを逐次的にコンテンツマネジメントサーバへ送信する．また，移動体は GPS などのポジショニングシステムにより得られた位置情報を，コンテンツマネジメントサーバへ送信する．登録されたコンテンツは，センサから取得したセンシングデータと，移動体から取得した位置情報を基に検索され，移動体へと返される．

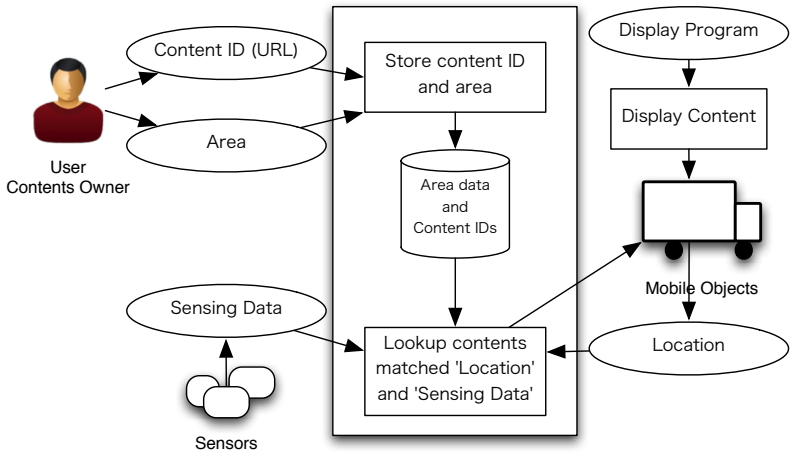


図 6.1: 実空間コンテキストに応じたコンテンツ配信システムの概要

6.2 システム動作設計

実空間コンテキストに応じたコンテンツ配信システムの動作について述べる．ここで，コンテンツの配信元が任意のコンテンツを配信先の移動端末に配信する場合を考える．また，コンテンツは時間コンテキストや物理コンテキストなどから構成される実空間コンテキスト条件に応じて配信されるとする．

いま，移動デジタルサイネージ環境を想定し，複数のコンテンツを移動端末で表示する場合を考える．このとき，コンテンツの配信方法は2通り考えられる．一つは，表示すべきコンテンツを配信先の要求に応じて配信を開始し，コンテンツを取得する必要がなくなるまで接続を継続して，必要に応じてコンテンツを配信する方法である．接続を継続する方法は，映像などの連続したデータを配信する際に効果的であるが，配信先の通信品質に合わせた配信を行う必要がある．一方で，配信先の移動端末がアクセスできるWebサーバにコンテンツを配置し，配信先がWebサーバへアクセスする方法で最新のコンテンツを配信できる．一般に，Webコンテンツは後者の配信方式を想定しており，URLを用いることでWebコンテンツへアクセスできる．移動デジタルサイネージ環境において，即時性を高めることにより，コンテンツの価値が下がる問題に対応することが望ましいため，コンテンツ配信方法は移動端末によりWebサーバへアクセスする方法を用いる．

実空間コンテキストに応じた配信を行うための実空間コンテキストの解析について考える．実空間コンテキストの解析は，移動体の位置やセンシングデータなどの物理コンテキストや，時間コンテキストから，コンテンツを配信する条件である実空間コンテキスト条件を検索することである．実空間コンテキスト条件を検索するには，センサより送信されるセンシングデータ，移動体の位置情報などの物理コンテキスト，そして実空間コンテキスト条件と対応するコンテンツを扱う必要がある．よって，配信元と配信先，そしてセンサノードからアクセスできる必要がある．そこで，コンテンツマネージメントサーバを構築し，そこに実空間コンテキストの情報を集めて実空間コンテキストの解析を行う．URLを用いてWebコンテンツへアクセスする方式を用い，コンテンツマネージメントサーバ上にコンテンツを配置することにより，システムとしての拡張性を確保する．

コンテンツの配信先の移動体は実空間コンテキストに応じて表示する情報を切り替える．いま，コンテンツマネージメントサーバにより実空間コンテキストが解析される．そのため，移動体に関連する物理コンテキストをコンテンツマネージメントサーバへ送信する必要がある．また，表示をするコンテンツへアクセスする必要がある．アクセスすべきコンテンツは実空間コンテキストの解析結果により異なる．そこで，取得すべきコンテンツのID（URL）をコンテンツマネージメントサーバに要求し，その結果得られるURLを用いてコンテンツが配置されたWebサーバへアクセスするものとする．取得すべきコンテンツは複数設定できるのが望ましいことを踏まえると，コンテンツマネージメントサーバに要求した結果は複数のURLからなるリストであることが望ましい．それぞれのURLで示されたコンテンツは時間コンテキストに応じて情報の切り替えるために，表示する順序と時間を記述できるようにする．

システム動作の設計をまとめると次の 3 つの機能が必要である．

コンテンツ登録
コンテンツ配信元によるコンテンツの登録と，コンテンツを配信するための条件である実空間コンテキスト条件の登録機能．

実空間コンテキストの解析
コンテンツマネージメントサーバによる実空間コンテキストから配信するための条件の検索機能．

コンテンツ取得　移動端末による取得すべきコンテンツ ID（URL）リストの取得要求と，Web サーバへのアクセス機能．

上記の機能を実現するため，コンテンツマネージメントサーバで AR-tree を用いた実空間コンテキストの解析を行い，得られた結果から移動体においてコンテンツを非同期にプリフェッチする．これらの機能を連携することにより，実空間コンテキストに応じたコンテンツ配信において，効率的な実空間コンテキストの解析と，最新の状態のコンテンツを安定的に取得する．

6.3 ネットワーク構成

実空間コンテキストに応じたコンテンツ配信システムにおける接続環境を述べる．ここで，移動端末上で動作するコンテンツを取得するためのアプリケーションを WebCrawler とする．図 6.2 は，WebCrawler と AR-tree が動作するシステムの接続構成を示している．コンテンツの配信元はホームネットワークを介してインターネットにアクセスし，コンテンツ配信システムへ情報を登録する．移動体とセンサは広域無線ネットワークを介してインターネットにアクセスし，コンテンツ配信システムから情報を取得する．画面を切り替えながら表示する実装である．AR-tree はコンテンツ配信サーバで動作し，WebCrawler は移動体に設置された情報端末で動作する．AR-tree は位置とセンシングデータに応じて，複数のコンテンツを検索する実装であり，コンテンツ配信における検索機能を持つ．WebCrawler は複数の Web を，予め用意されたスケジュールに従って情報を表示する実装であり，スケジュールに影響しないようコンテンツを最新に保つための取得機能を持つ．AR-tree と WebCrawler が連携する事により，配信元が登録したコンテンツは，最新の状態で移動体へ配信できる．

6.4 まとめ

実空間コンテキストに応じたコンテンツの配信は，コンテンツの登録と実空間コンテキストの解析，そしてコンテンツの取得を連携することにより実現する．コンテンツの登録時には，コンテンツ ID（URL）とコンテンツを配信するための実空間コンテキスト条件を登録する．コンテンツの取得時には，移動端末の現在位置をコンテンツマ

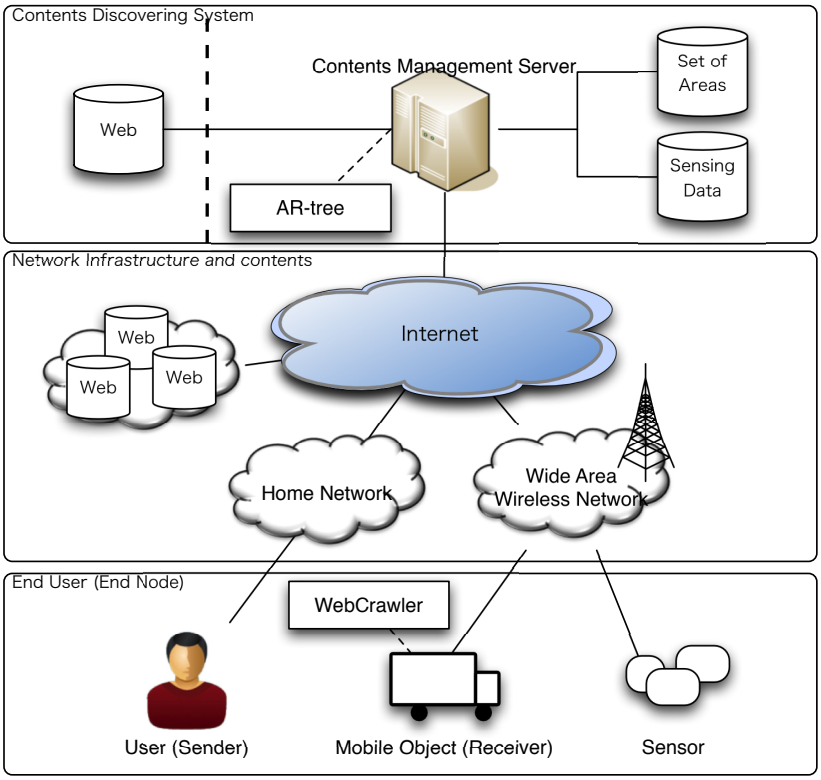


図 6.2: 実空間情報に応じた情報配信システムの接続構成図

ネージメントサーバへ送信し，結果として得られるコンテンツ ID リストを基に，Web サーバへアクセスする．取得したコンテンツは，表示スケジュールに応じて情報端末に表示される．コンテンツを配信するための実空間コンテキスト条件は，コンテンツ登録時の配信対象の領域と，センサにより定期的に得られる実空間コンテキストにより構成される．

第7章　実装

第 6 章の設計に基づき，AR-tree と WebCrawler を実装した．AR-tree は実空間コンテキスト条件をインプットデータとし，実空間コンテキストを検索するためのアルゴリズムである．WebCrawler はコンテンツを任意の時刻で取得し表示するアプリケーションである．AR-tree と WebCrawler を連携することにより，実空間コンテキストに応じたコンテンツ配信を実現する．

7.1　AR-tree の実装

実空間コンテキストを検索するためのプロトタイプとして **AR-tree** を実装した．AR-tree は，コンテンツマネージメントサーバにおいて実行されることを想定した Java アプリケーションである．AR-tree は実空間コンテキスト条件をインプットデータとし，実空間コンテキストを検索する．プロトタイプとして実装したため，実空間コンテキスト条件を 2 次元空間における領域データ，時々観測される実空間コンテキストを点のデータと見なし，点のデータにより領域データを検索する機能を実装した．

以下に実装したクラスを挙げ，それぞれの機能を説明する（クラス図は D.1 を参照）．

ARtree

AR-tree を表すクラスである．プロパティとして ARTreeIndex 型の Root 変数を持つ．主な機能として次に挙げるメソッドを実装する．

appendArea

領域データの新規に追加する．データノードに領域データを追加すると共に，AR-tree のインデクスを新規作成し適切な場所に挿入する．

searchArea

点のデータにより領域データを検索する．実行結果として点のデータを含む領域データを返す．

ARtreeArea

領域を表すデータクラスである．AR-tree へインデクスを追加するための領域比較メソッドを持つ，領域比較メソッドは比較の結果として，任意の 2 つの領域の関係を返す．$(INTERSECT, DISJOINT, INSIDE, CONTAIN)$

ARtreePoint

点を表すデータクラスである．

ARtreeNode
AR-tree のノードを抽象的に表すクラスである．ARtreeIndex と ARtreeLeaf クラスのスーパークラスである．

ARtreeIndex
AR-tree のインデクスノードを表すクラスである．プロパティとして ARtreeArea 型の領域データを持つ．

ARtreeLeaf
AR-tree のデータノードを表すクラスである．インプットデータの冪集合となる ARtreeArea 型の領域集合を持つ．

ARtreeQuery
AR-tree の検索データセットを表すクラスである．ARtreePoint クラスの変数リストである．

ARtreeData
AR-tree のデータセットを表すクラスである．ARtreeArea クラスの変数リストである．

7.2 WebCrawler の実装

非同期プリフェッチを用いてコンテンツを表示する **WebCrawler** を実装した．WebCrawler は巡回バス車内の PC において実行される，Adobe AIR[20] アプリケーションである．WiMAX 網を利用してインターネットに接続し，インターネットを通じてコンテンツを取得する．Adobe AIR は Rich Internet Application(RIA) に分類される Web アプリケーション実行環境であり，非同期に Web コンテンツを取得することを考慮している．WebCrawler を実装した PC はバス車内に設置された WiMAX 端末と Ethernet ケーブルで接続した．

図 7.1 に WebCrawler の動作概要を示す．WebCrawler は非同期プリフェッチを実装するため，**コンテンツ表示部**と**コンテンツ取得部**に機能を分割した．コンテンツ表示部はスケジュールされたコンテンツ表示時間に従って，表示するページの切り替えを行う．コンテンツ取得部は取得時刻のスケジュールに従って，コンテンツを取得する．コンテンツ表示部とコンテンツ取得部は，各ページ番号のコンテンツデータ群を共有して連携する．各コンテンツデータは画面への表示用と蓄積用の領域を持つ．取得中のコンテンツデータは蓄積用の領域に蓄えられ，取得が完了したら表示用のデータ領域と入れ替える．コンテンツ表示時間のスケジュールは，設定ファイルにより記述された時間を巡回するようにし，コンテンツ取得時刻のスケジュールは，表示スケジュールにおけるコンテンツのページ番号を 1 減算したものを用いるよう実装した（WebCrawler のスケジュール記述ルールは付録 E を参照）．

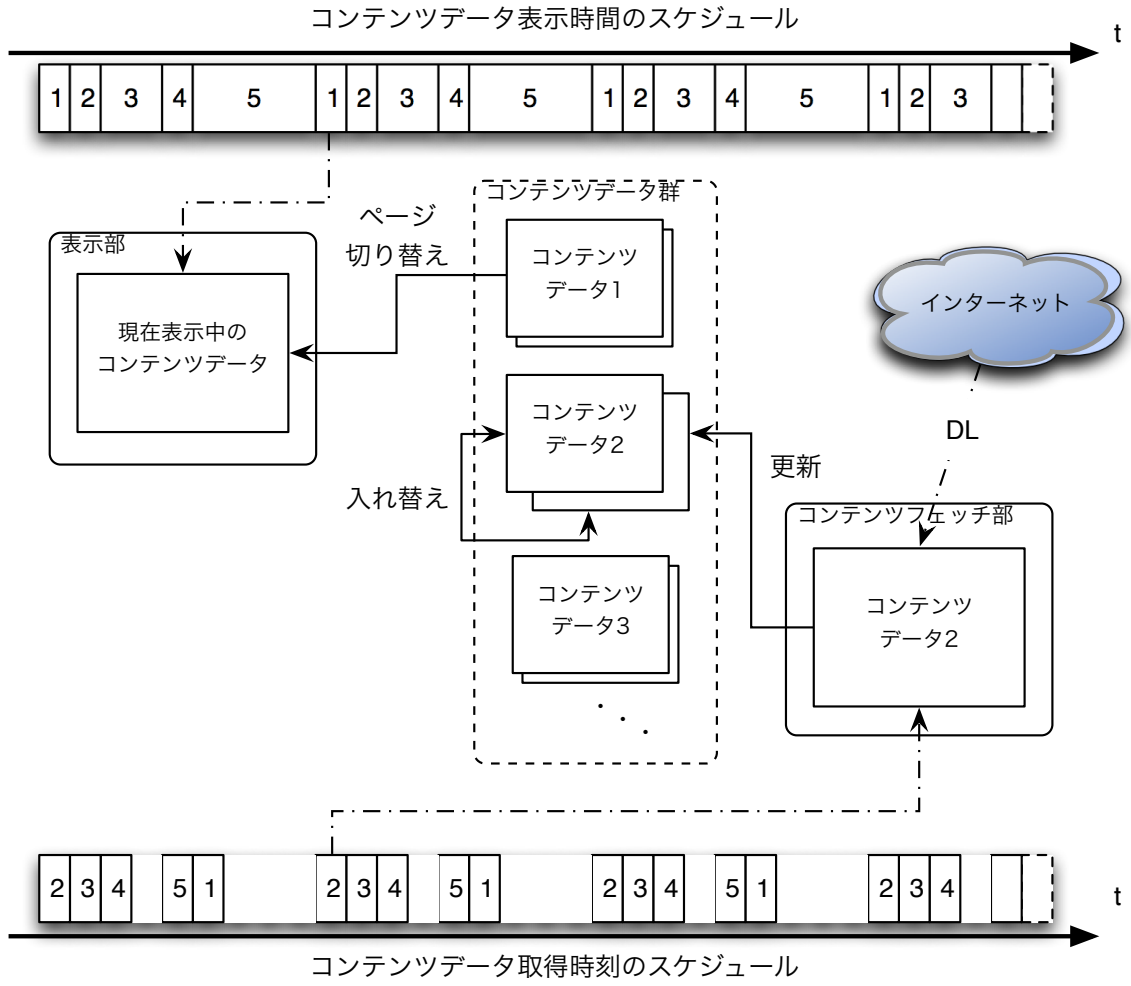


図 7.1: WebCrawler の動作概要

第8章　評価

AR-tree，WebCrawler により実現した提案手法を評価する．AR-tree は条件集合から効率的な検索するための条件検索手順を示すアルゴリズムであり，条件間の関係により検索効率が異なる．そこで，条件間の関係性に異なる特徴を持つ 5 種類のデータセットを用いて測定した検索時間により評価する．また，検索効率を高めるために要した処理時間（条件の追加にかかる時間）を測定し評価する．WebCrawler は非同期プリフェッチにより移動体における安定的な情報取得を実現するアプリケーションである．そこで，コンテンツの取得開始時刻から表示開始時刻までの時間により，非同期プリフェッチによるコンテンツ取得時間の安定性を評価する．コンテンツの取得時間は移動デジタルサイネージ環境において実測する．

8.1　実空間コンテンツ条件の検索時間による評価

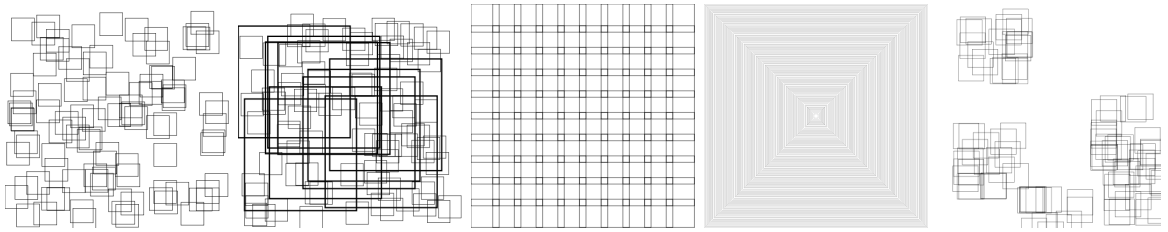
AR-tree を用い，実空間コンテキスト条件の検索時間による検索効率を評価する．Java VM が動作する Mac OS X の PC（CPU は 2GHz が 2 コア，Memory は 2GB 667MHz）において，YourKit Java Profiler[21] を用いて測定した（表 8.5）．

CPU	2GHz x 2 コア
Memory	2GB 667MHz
OS	Mac OS X 10.5.8
Java VM	JVM 1.5
Profiler	YourKit Java Profiler 8.0.20

表 8.1: AR-tree の条件比較速度測定環境

検索効率は対象とする条件集合における条件間の関係により変わる．条件を空間データにおける範囲として考えたとき，条件間の関係は，空間における範囲のサイズと分布の特徴により特徴づけられる．ここで，条件を空間データにおける範囲とし，分布に特徴を持つ空間データセットを用いて効率の変化を測定した．検索効率は検索に必要となった領域の比較回数により評価する．図 8.1 に測定に用いた 5 種類の空間データセットを示し，各データセットの分布の特徴を次に述べる．

UNIFORM (領域の数：100 個，領域のサイズ：1)
　固定サイズの領域が二次元の空間に一様に分布しているデータである．



(a) UNIFORM (b) MIX (c) PARCEL (d) CONCENT (e) CLUSTER

図 8.1: AR-tree の評価に用いたデータセット

MIX (領域の数：100 個，領域のサイズ：1 と 5 の二種類 (1：90 個, 5：10 個))
二種類のサイズの領域が二次元の空間に一様に分布しているデータである．二次元の空間に一様に分布しているデータを，サイズが 1 である領域を 90 個，サイズが 5 である領域を 10 個作成し，それらをまとめて一つのデータセットにした．

PARCEL (領域の数：100 個，領域のサイズ：1.3，領域の距離：1)
固定サイズの領域が一定間隔を空けて交差して分布しているデータである．格子状に分布する領域を，隣の領域と交差するよう拡大して作成した．

CONCENT (領域の数：100 個，領域のサイズ：100 種類)
階層的に領域同士が包含されるように分布しているデータである．複数サイズの領域の中心が固定されるように作成した．

CLUSTER (領域の数：100 個，領域のサイズ：1，局所点：5 箇所)
5 箇所の局所点を二次元空間に一様に分布させ，それぞれの局所点を中心として，20 個の領域が同心円内に一様に分布するようにして作成した．

AR-tree による検索効率を測定した結果を表 8.2 にまとめる（詳細は付録 F.1–F.5 を参照）．表 8.2 より，CONCENT，MIX において，検索時間の効率化されていることが平均比較回数 (Avg) より分かる．これは CONCNET と MIX が比較的多くの包含関係を持つためであると考えられる．MIX においては，検索クエリによらず効率化されていることが標準偏差 (StdDev) の値 3.97 から読み取れる．CONCENT においては，検索クエリにより，良く効率化されている場合と，あまり効率化されていない場合があることが標準偏差の値 22.96 から読み取れる．これは最も多くの比較が必要である場合（集合全ての条件に一致しない）が起こることで標準偏差に差が出たと，2 つの最大比較回数（Max）の値により考えられる．一方，空間データの追加操作は，空間データの数 100 を超える場合が存在した．これは，空間データ同士が交差している場合に，ツリーにおけるノードの追加を再帰的に行ったためである．以上の結果より，AR-tree により検索効率が全体的に向上させることができていると言える．

既存手法の中で最も普及している手法の一つである R-tree と比較することで，AR-tree による検索効率の優位性を示す．R-tree はツリーにおける一つの節は複数のインデクスデータを持つ．インデクスデータの最大値は Page size により定まり，この値

	AR-tree							
	検索				追加			
	比較回数			時間 (ms)	比較回数			時間 (ms)
	Avg	Max	StdDev	Avg	Avg	Max	StdDev	Avg
UNIFORM	63.21	100	34.32	-	63.21	100	34.32	-
MIX	21.59	24	3.97	0.11	21.59	24	3.97	0.11
PARCEL	44.90	100	29.63	0.01	44.90	100	29.63	0.01
CONCENT	9.80	101	22.96	0.09	9.80	101	22.96	0.09
CLUSTER	71.74	92	28.63	0.08	71.74	92	28.63	0.08

表 8.2: AR-tree の検索速度と追加速度

により効率が変わる．いま，Page size を標準値である 1024bytes を始め，512bytes，256bytes，128bytes の 4 つの値を用いて，検索効率を測定した．測定に用いた R-tree は R-tree Portal[22] に公開されている，Timos Sellis 氏による C 言語で開発された実装である．検索効率は AR-tree と同様に，検索に必要とした比較回数の平均値により評価し，AR-tree における比較回数の平均値を 100 とした相対値により示す（表 8.3，図 8.2）．

	R-tree 検索における比較回数（相対値）				
	UNIFORM	MIX	PARCEL	CONCENT	CLUSTER
Page size 1024bytes	161	741	345	測定不可[1]	141
Page size 512bytes	112	533	327	測定不可[1]	99
Page size 256bytes	97	443	181	測定不可[1]	86
Page size 128bytes	72	439	138	測定不可[1]	63

表 8.3: R-tree の検索時間（AR-tree の平均比較回数を 100 とした相対値）

	R-tree 検索における比較回数（標準偏差）				
	UNIFORM	MIX	PARCEL	CONCENT	CLUSTER
Page size 1024bytes	39.02	33.91	33.54	測定不可[1]	39.02
Page size 512bytes	37.43	34.62	71.87	測定不可[1]	37.43
Page size 256bytes	41.11	46.46	39.43	測定不可[1]	41.11
Page size 128bytes	39.05	49.48	37.98	測定不可[1]	39.05

表 8.4: R-tree の検索時間（比較回数の標準偏差）

R-tree の検索効率を測定した結果より，UNIFORM データセットを用いて，Page size を 256bytes，128bytes にした場合と，CLUSTER データセットを用いて，Page size を 512bytes，256bytes，128bytes にした場合を除き，AR-tree がより高い検索効率であることが分かった．AR-tree と R-tree を比較した結果，AR-tree が R-tree より高い検索効

[1]用いた実装が正常に動作を完了しなかったため計測できなかった

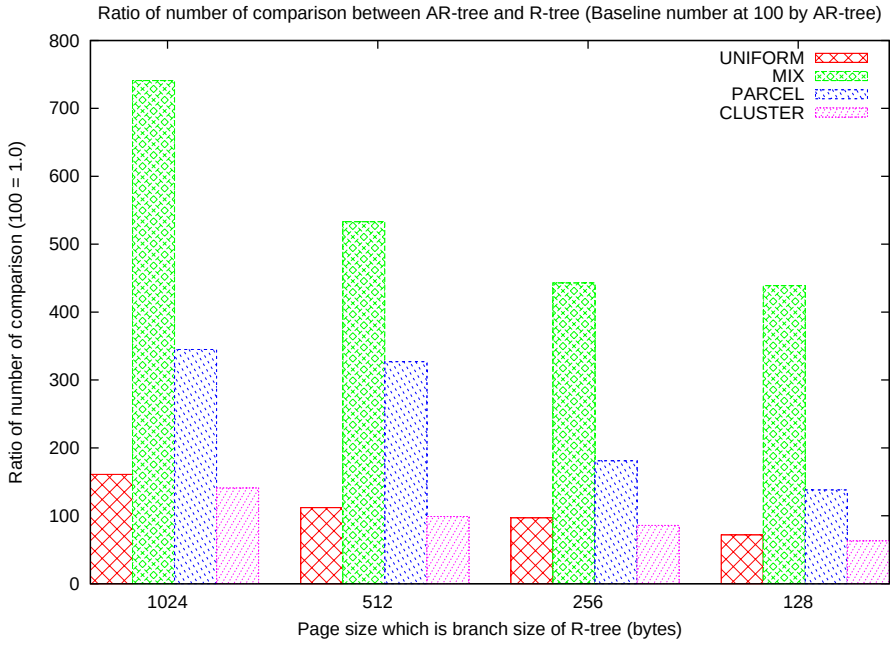


図 8.2: R-tree の検索時間（AR-tree の平均比較回数を 100 とした相対値）

率を示せたのは，Page size がいずれの場合においても MIX データセットを用いた場合であった．一方，R-tree より高い検索効率を示せなかったのは，Page size がいずれの場合においても CLUSTER データセットを用いた場合であった．この結果は，R-tree が領域間の距離を元にインデクスを構成するため，局所的に点在するデータに対して高い効率を示せたものと考えられる．同様に，複数の領域をまたがる領域が存在する MIX に近い分布のデータに対しては，R-tree は効率化できていないと考えられる．

以上の結果より，データセットに違いによらず多くの場合において AR-tree は R-tree より高い検索効率を実現できたと言える．

8.2 コンテンツ表示までの所要時間の分布確率による評価

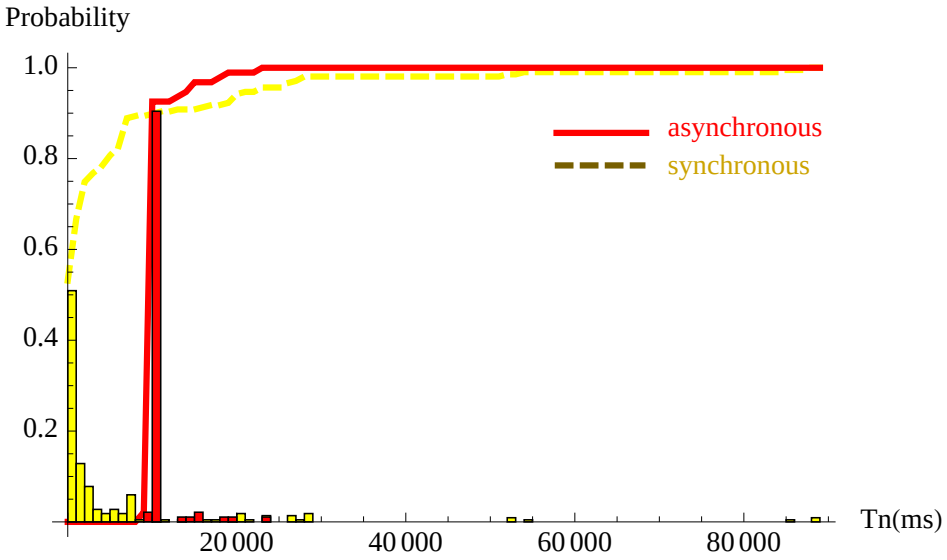
WebCrawler を用い，コンテンツの取得開始時刻から表示開始時刻までの時間 T_n により，コンテンツ取得時間の安定性を評価する．Adobe Air Runtime が動作する Windows XP の PC（CPU は 1.6GHz が 1 コア，Memory は 1GB 400Mhz）において，WebCrawler の動作ログを用いて測定した（表 8.5）．

移動デジタルサイネージ環境 2.2 において，比較的通信品質が悪い移動経路 A と，比較的通信品質がよい移動経路 B を走行し，同期フェッチと非同期プリフェッチの T_n の時間を計測した．計測時においてディスプレイに表示するコンテンツは，一定の 10 秒間隔で表示を切り替える設定を用いた．移動経路 A における測定結果を図 8.3 に，移動経路 B における測定結果を図 8.4 に示す．

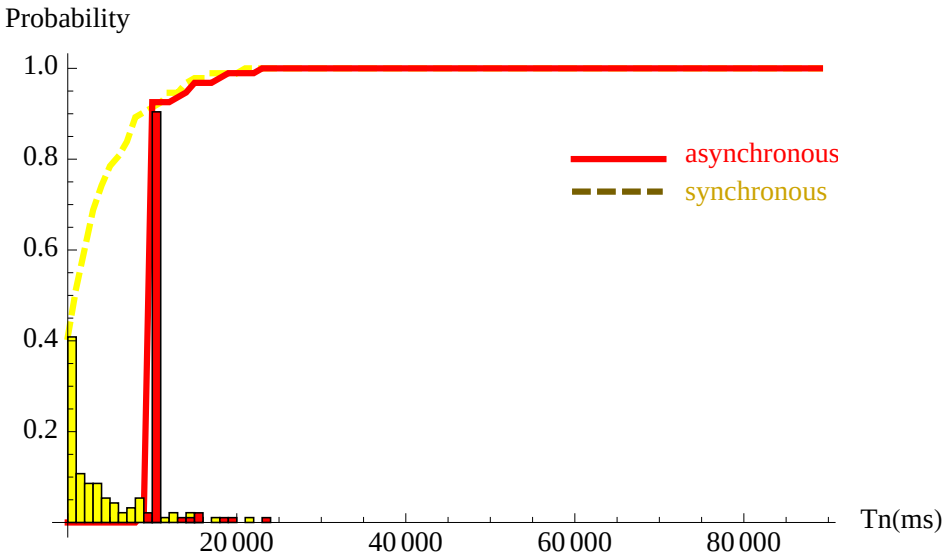
図 8.3，8.4 は，WebCrawler がコンテンツを取得開始してから表示を開始するまで時

CPU	1.6GHz x 1 コア
Memory	1GB 400MHz
OS	Windows XP
Adobe Air Runtime	version 1.5.2

表 8.5: WebCrawler の動作測定環境



(a) 試行 1



(b) 試行 2

図 8.3: 移動経路 A におけるコンテンツ表示までの所要時間の確率分布と累積確率分布

間 T_n の確率分布とその累積確率を，それぞれ同期フェッチと非同期プリフェッチの 2 つの場合について測定した結果である．通信品質が比較的少なく変動する移動経路 A において，同期フェッチと非同期プリフェッチの両方式は，コンテンツの表示時間 10

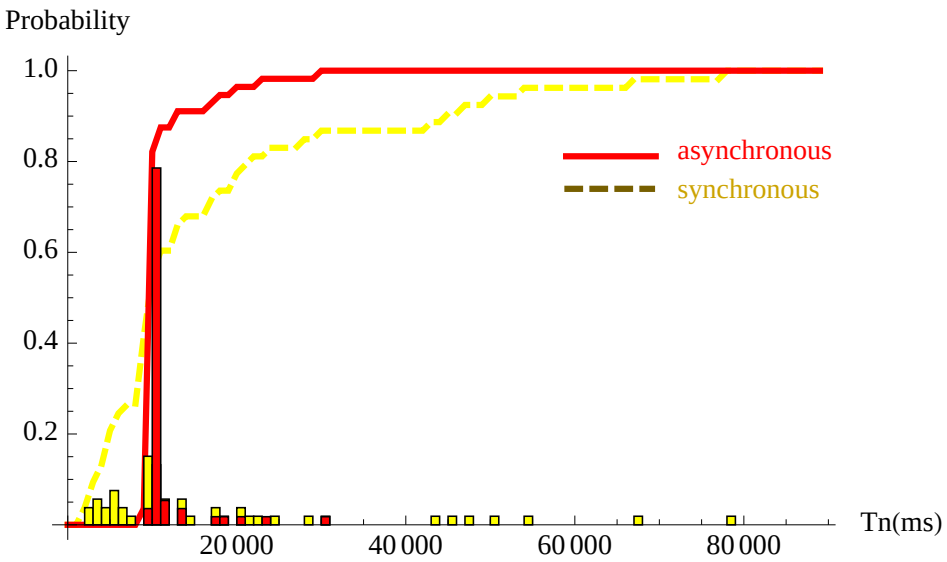


図 8.4: 移動経路 B におけるコンテンツ表示までの所要時間の確率分布と累積確率分布

秒を経過するまでに，累積確率の値がおよそ 0.9 となり安定して取得できた．一方，通信品質が比較的大きく変動する移動経路 B においては，同じくコンテンツの表示時間 10 秒を経過するまでに，累積確率の値が同期フェッチではおよそ 0.6 となり，非同期プリフェッチではおよそ 0.9 となり，同期フェッチに比べて安定してコンテンツを取得できた．

非同期プリフェッチは移動経路による通信品質の違いによらず安定してコンテンツを取得でき，通信品質が悪い場合において特に効果が期待できると言える．以上の結果により，非同期プリフェッチは同期フェッチと比較し，コンテンツを安定して取得することができたと言える．

第9章　結論

9.1　AR-treeによる実空間コンテキストの高速検索

AR-treeを用いることで，点のデータにより効率的に領域のデータを検索できた．特に，領域間に包含関係を多く持つデータセットに対して高い効果が示された．AR-treeは最も時間のかかる場合（多くは集合全ての条件に一致しない場合）の検索時間においても，オーダ$O(N)$を超えることはないと保証できる．AR-treeによる検索効率は，最も有名な既存研究の一つであるR-treeと比較し，多くの点で高い効率を示すことができた．領域のデータを実空間コンテキスト条件とし，点のデータを時々変化する実空間コンテキストに適応することにより，同様に効率的な検索が可能である．この特徴は従来研究にない利点であり，AR-treeの検索効率は既存研究に比べて優位である．

AR-treeを用いることにより，移動デジタルサイネージ環境において，効率的に実空間コンテキストを解析できる．従来方式に比べて少ない時間で実空間コンテキストを解析できるため，比較的頻度が高く変化する実空間コンテキストを処理できる．

9.2　非同期プリフェッチによる情報の安定取得

非同期プリフェッチは，コンテンツを表示する前に取得する同期フェッチに比べ，コンテンツデータを取得してから表示するまでにかかる時間の変動を抑え，コンテンツを安定して取得できる．確率分布の最大値がほぼ二倍になることは，コンテンツデータを取得するための時間を十分に確保でき，安定的にコンテンツ表示できることにつながる．この特徴は通信品質が比較的小さく変動する場合でもわずかながらに見られた．また，通信品質が比較的大きく変動する場合では高い効果を示した．通信品質が大きく変動する移動環境において，非同期プリフェッチの効果は高い．

非同期プリフェッチを用いることにより，移動デジタルサイネージ環境において，安定的にコンテンツを表示できた．従来方式でのネットワークベースでのプリフェッチや，ネットワークにおけるキャッシュ配置，移動体の位置情報による効率改善では，短い時間間隔で最新の情報を取得することが困難であったため，交通情報やブログのようなリアルタイム性を要求されるWebコンテンツを安定的に表示することが難しい．しかし，非同期プリフェッチはコンテンツの取得順序を表示順序に依らずスケジューリングすることができるため，リアルタイム性を要求されるWebコンテンツを細かい粒度で安定的に表示できると期待される．

付 録A　センシングデータ量予測環境

図 A.1 は実空間コンテキストの解析を行うシステムで必要とするデータ処理能力を予想する際に想定した，センシングデータの取得環境を示す．一般的なセンサ利用環境に基づき，センサノードやセンサネットワークを適宜分散して配置した．

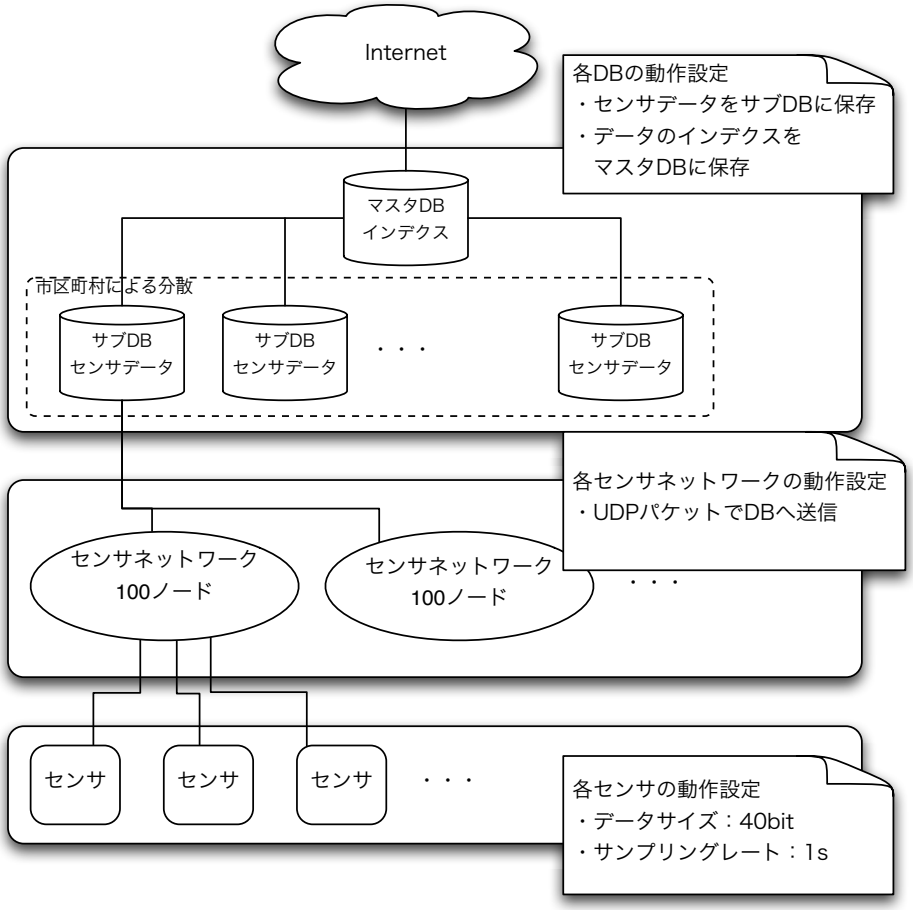
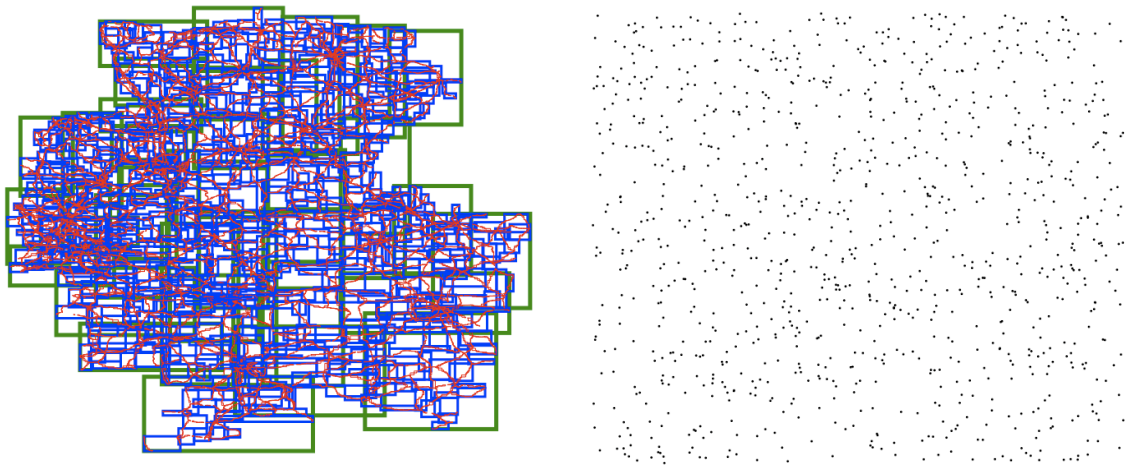


図 A.1: センシングデータ量予測に用いた動作設定

付 録B　階層構造による規模性測定に用いたデータセット

図 B.1 は，第 3.3 節で述べた階層構造による規模性測定に用いたデータセットを示す．図 B はデータセットに対する階層的なインデクスを構成した例を示す．インデクスと対象データが共に示される．図 B はデータセットを検索するためのクエリデータを示す．



(a) 対象データとインデクス　　(b) クエリデータ

図 B.1: 階層構造による規模性測定に用いたデータ

付 録C　AR-treeにおける領域比較手順の簡易化

AR-tree への領域の追加において，領域の比較を簡易化するための手法を示す．以下は，ツリーの IndexNode が持つ領域とツリーへ追加する領域の関係より，求める領域との関係が導けることを証明する．ここで，ツリーへ追加する領域 NewArea を N，追加する対象の AR-tree における任意の IndexNode を A_n，A_n の親から Root までの IndexNode の領域を結合した領域を P_n．P_n と A_n を結合した領域を I_n．P_n と $\overline{A_n}$ を結合した領域を O_n とする．いま，領域 N をツリーに追加する問題は，領域 P_n を全集合とする領域を $P_n \cap N$ と $P_n \cap A_n$ の 2 つの領域による新たな領域作成の問題と考えられる，このとき，$P_n \cap N$ と $P_n \cap A_n$ の関係の一部は，N と A_n の関係より定まることを以下に示す．

まず，A_n がツリーの Root である場合，$P_n = 1$ であり，$P_n \cap A_n = A_n$，$P_n \cap N = N$ となる．従って，A_n がツリーの Root である場合，$P_n \cap A_n$ と $P_n \cap N$ の関係は A_n と N の関係と等しい．

次に，A_n がサブツリーの最上位である場合（$P_n \neq 1$）を考える．A_n と N は，次に示す 4 つのいずれか関係を持つ．

1. $A_n \supseteq N$
2. $A_n \subseteq N$
3. $A_n \cap N = \emptyset$
4. $A_n \cap N \neq \emptyset$

以下，A_n と N の関係により場合を分けて考える．

A_n と N の関係が 1 である場合，$A_n \cap \overline{N} = \emptyset$ である．すると，$P_n \cap N \subseteq P_n \cap A_n$ の条件（$(P_n \cap A_n) \cap (\overline{P_n \cap N}) = \emptyset$）を以下の通り満たす．

$$\begin{aligned}(P_n \cap A_n) \cap (\overline{P_n \cap N}) &= (P_n \cap A_n) \cap (\overline{P_n} \cup \overline{N}) \\ &= (P_n \cap A_n \cap \overline{N}) \cup (P_n \cap A_n \cap \overline{P_n}) \\ &= \emptyset\end{aligned}$$

従って，$A_n \subseteq N$ のとき $P_n \cap N \subseteq P_n \cap A_n$ を満たす．

A_n と N の関係が 2 である場合，$\overline{A_n} \cap N = \emptyset$ である．すると，$P_n \cap N \supseteq P_n \cap A_n$ の条件（$(\overline{P_n \cap A_n}) \cap (P_n \cap N) = \emptyset$）を以下の通り満たす．

$$\begin{aligned}(\overline{P_n \cap A_n}) \cap (P_n \cap N) &= (\overline{P_n} \cup \overline{A_n}) \cap (P_n \cap N) \\ &= (\overline{P_n} \cap \overline{A_n} \cap P_n) \cup (\overline{P_n} \cap \overline{A_n} \cap N) \\ &= \emptyset\end{aligned}$$

従って，$A_n \supseteq N$ のとき $P_n \cap N \supseteq P_n \cap A_n$ を満たす．

A_n と N の関係が 3 である場合，$A_n \cap N = \emptyset$ である．すると，$P_n \cap N \cap P_n \cap A_n = \emptyset$ を以下の通り満たす．

$$\begin{aligned}(P_n \cap N) \cap (P_n \cap A_n) &= P_n \cup N \cap A_n \\ &= \emptyset\end{aligned}$$

従って，$A_n \cap N = \emptyset$ のとき $P_n \cap N \cap P_n \cap A_n = \emptyset$ を満たす．

A_n と N の関係が 4 である場合，A_n と N の関係より $P_n \cap A_n$ と $P_n \cap N$ の関係を示せない．

以上をまとめると，A_n と N の関係より以下が導ける．

$$\begin{cases} P_n \cap N \subseteq P_n \cap A_n & (A_n \subseteq N \text{ のとき}) \\ P_n \cap N \supseteq P_n \cap A_n & (A_n \subseteq N \text{ のとき}) \\ P_n \cap N \cap P_n \cap A_n = \emptyset & (A_n \cap N = \emptyset \text{ のとき}) \\ (P_n \cap N) \text{ と } (P_n \cap A_n) \text{ の関係は不定} & (A_n \cap N \neq \emptyset \text{ のとき}) \end{cases}$$

同様に，A_n と P_n，P_n と N の関係より，以下が導ける．

$$\begin{cases} P_n \cap N \supseteq P_n \cap A_n & (A_n \supseteq P_n \text{のとき}) \\ (P_n \cap N) \text{ と } (P_n \cap A_n) \text{ の関係は不定} & (\text{上記以外のとき}) \end{cases}$$

$$\begin{cases} P_n \cap N \supseteq P_n \cap A_n & (P_n \subseteq N \text{ のとき}) \\ P_n \cap N \cap P_n \cap A_n = \emptyset & (P_n \cap N = \emptyset \text{ のとき}) \\ (P_n \cap N) \text{ と } (P_n \cap A_n) \text{ の関係は不定} & (\text{上記以外のとき}) \end{cases}$$

従って，条件が交差する場合を除き，A_n と N の関係や A_n と P_n，P_n と N の関係より，$P_n \cap A_n$ と $P_n \cap N$ の関係の一部を導ける．これにより，領域の追加操作において，領域比較を簡易的に行うことができる．

付 録D　AR-treeのクラス図

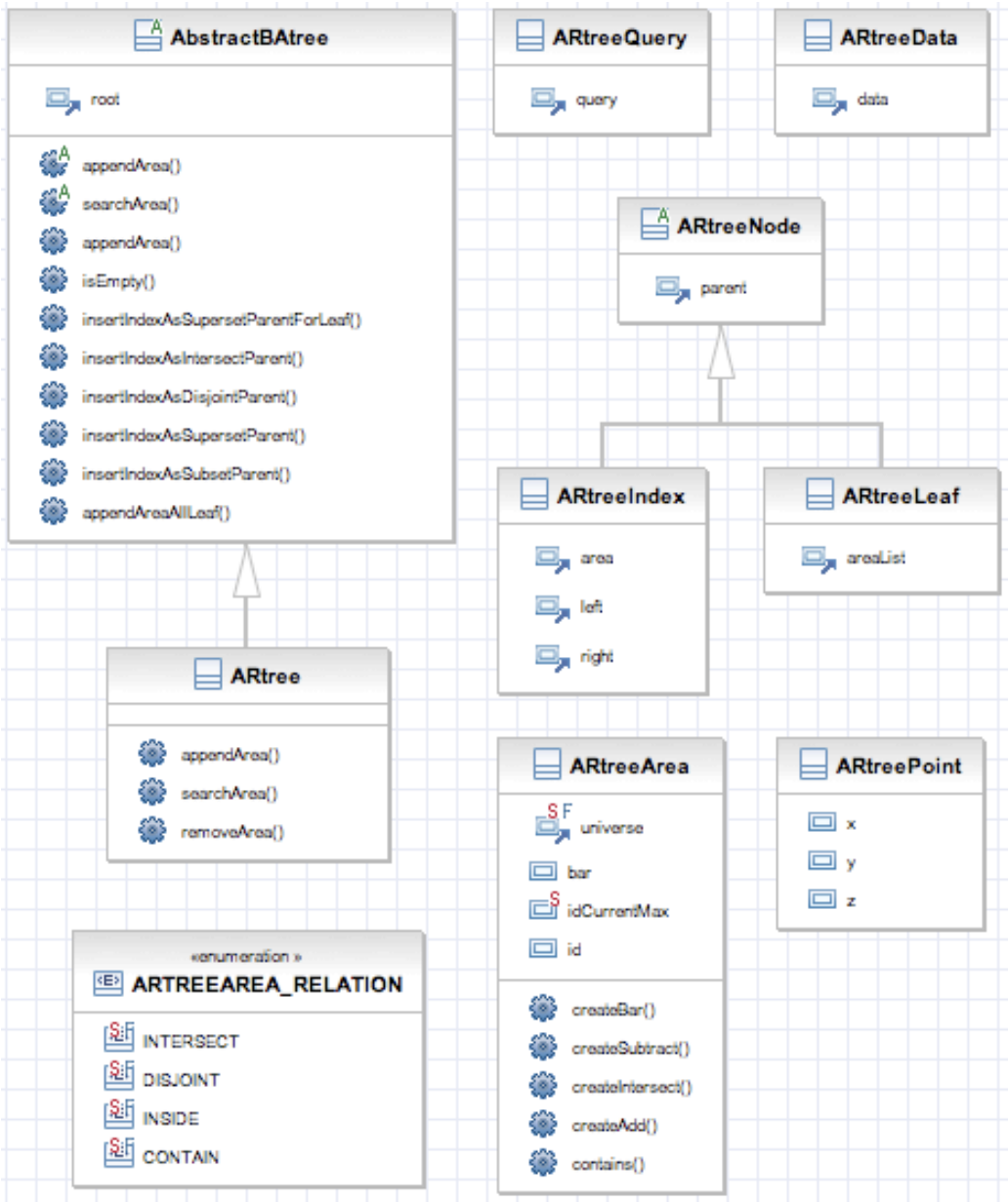


図 D.1: AR-tree のクラス図

付 録E　WebCrawlerの スケジュール記述ルール

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="WebCrawler">

    <!-- ページの内容 -->
    <xsd:element name="weblist">
      <xsd:complexType>

        <xsd:sequence>
          <xsd:element ref="twitter"/>  <!-- Twitter ページ -->
          <xsd:element ref="html"/>     <!-- HTML ページ -->
          <xsd:element ref="rss"/>      <!-- RSS ページ -->
        </xsd:sequence>

        <xsd:attribute name="autoselect" type="xsd:boolean"/> <!-- 自動ページめくりの有効化/無効化 -->
      </xsd:complexType>
    </xsd:element>
  </xsd:element>

  <!-- Twitter ページ -->
  <xsd:element name="twitter">
    <xsd:complexType>

      <xsd:sequence>
        <xsd:element name="public_timeline"/>  <!-- public timeline ページ -->
        <xsd:element name="user_timeline">     <!-- user timeline ページ -->
          <xsd:attribute name="id" type="xsd:string"/>  <!-- ユーザ ID -->
        </xsd:element>
      </xsd:sequence>

      <xsd:attribute name="displayinterval" type="xsd:nonNegativeInteger"/> <!-- 表示時間 -->
      <xsd:attribute name="crawlinterval" type="xsd:nonNegativeInteger"/>   <!-- 更新時間 -->
      <xsd:attribute name="maxtimelinenum" type="xsd:nonNegativeInteger"/>  <!-- 最大表示 Tweet 数 -->
      <xsd:attribute name="maxtextlength" type="xsd:nonNegativeInteger"/>   <!-- 最大表示文字数（超過文字
は省略（...）表示） -->
    </xsd:complexType>
  </xsd:element>

  <!-- HTML ページ -->
  <xsd:element name="html">
    <xsd:complexType>
      <xsd:attribute name="url" type="xsd:string"/>                         <!-- URL -->
      <xsd:attribute name="displayinterval" type="xsd:nonNegativeInteger"/> <!-- 表示時間 -->
      <xsd:attribute name="crawlinterval" type="xsd:nonNegativeInteger"/>   <!-- 更新時間 -->
    </xsd:complexType>
  </xsd:element>

  <!-- RSS ページ -->
  <xsd:element name="rss">
```

```
    <xsd:complexType>
      <xsd:attribute name="url" type="xsd:string"/>                          <!-- URL -->
      <xsd:attribute name="displayinterval" type="xsd:nonNegativeInteger"/> <!-- 表示時間 -->
      <xsd:attribute name="crawlinterval" type="xsd:nonNegativeInteger"/>   <!-- 更新時間 -->
      <xsd:attribute name="maxrssitem" type="xsd:nonNegativeInteger"/>      <!-- 最大表示項目数 -->
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

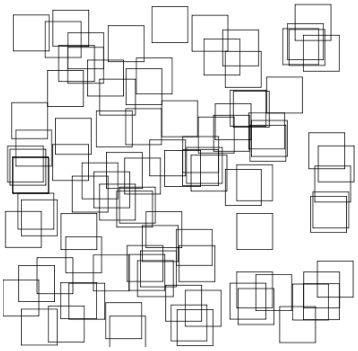
付 録F　AR-treeの検索効率測定データ

図 F.1–図 F.5 は AR-tree の検索効率を 5 種類のデータセットを用いて計測した結果である．各結果において次に述べる対象データ (a)，検索ツリーの深さ (b)，領域比較回数 (c) を示した図を示す．

対象データ 検索対象のインプットデータセットを表す．

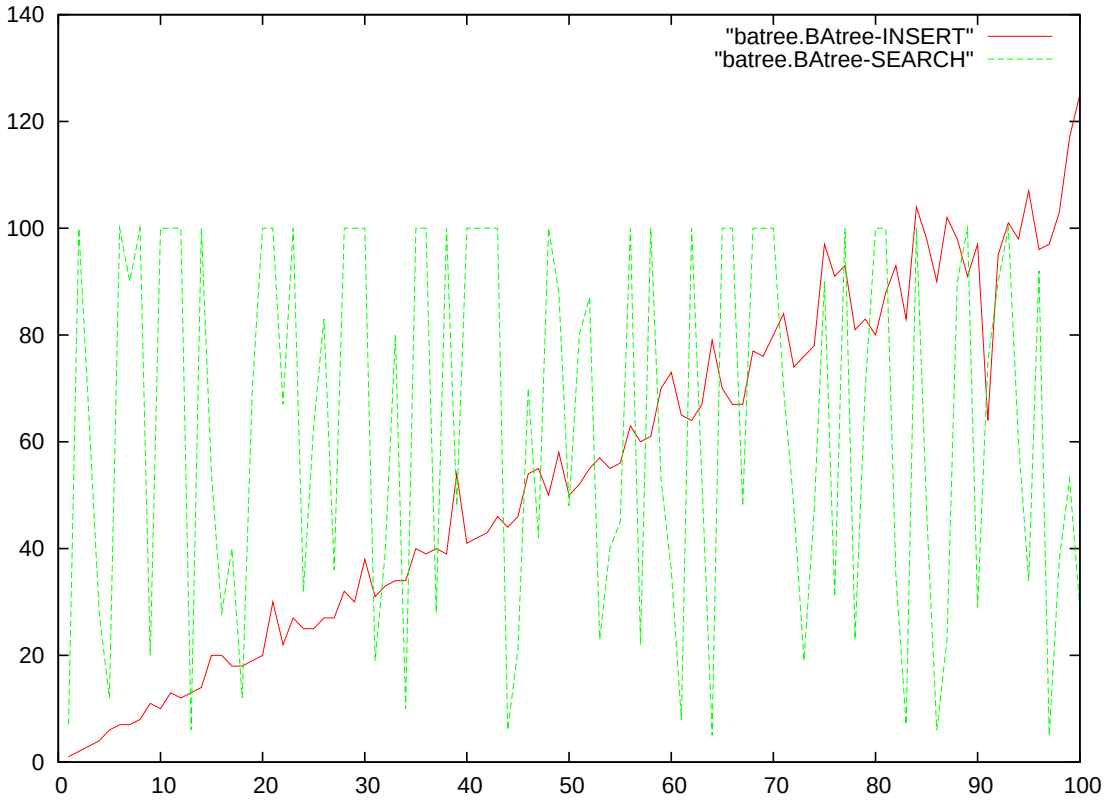
検索ツリーの深さ インプットデータを全て挿入した後のツリーにおけるデータノードの深さの分布を表す．データノードの深さは Root からデータノードまでのノード数とした．検索ツリーの深さで示した図は，縦軸を下方向を正にとり，一般的にツリー構造を示したときの Root が表される方向と合わせた．

領域比較回数 インプットデータの挿入と検索における領域の比較回数を表す．横軸にデータ番号をとり，縦軸に比較回数をとった．



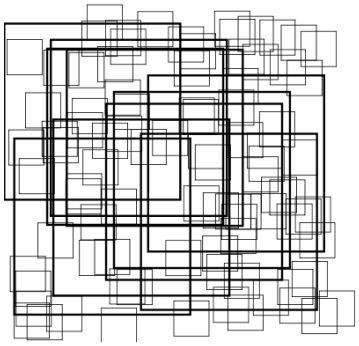
(a) 対象データ

(b) 検索ツリーの深さ

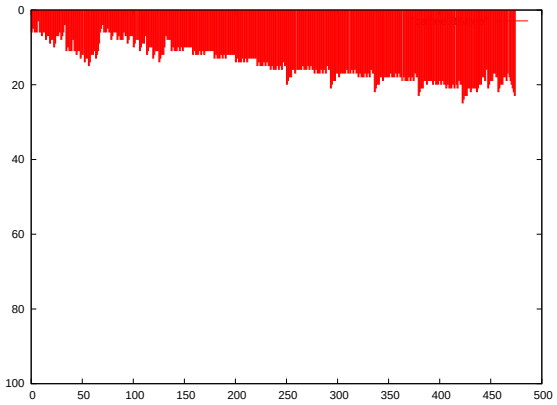


(c) 領域比較回数

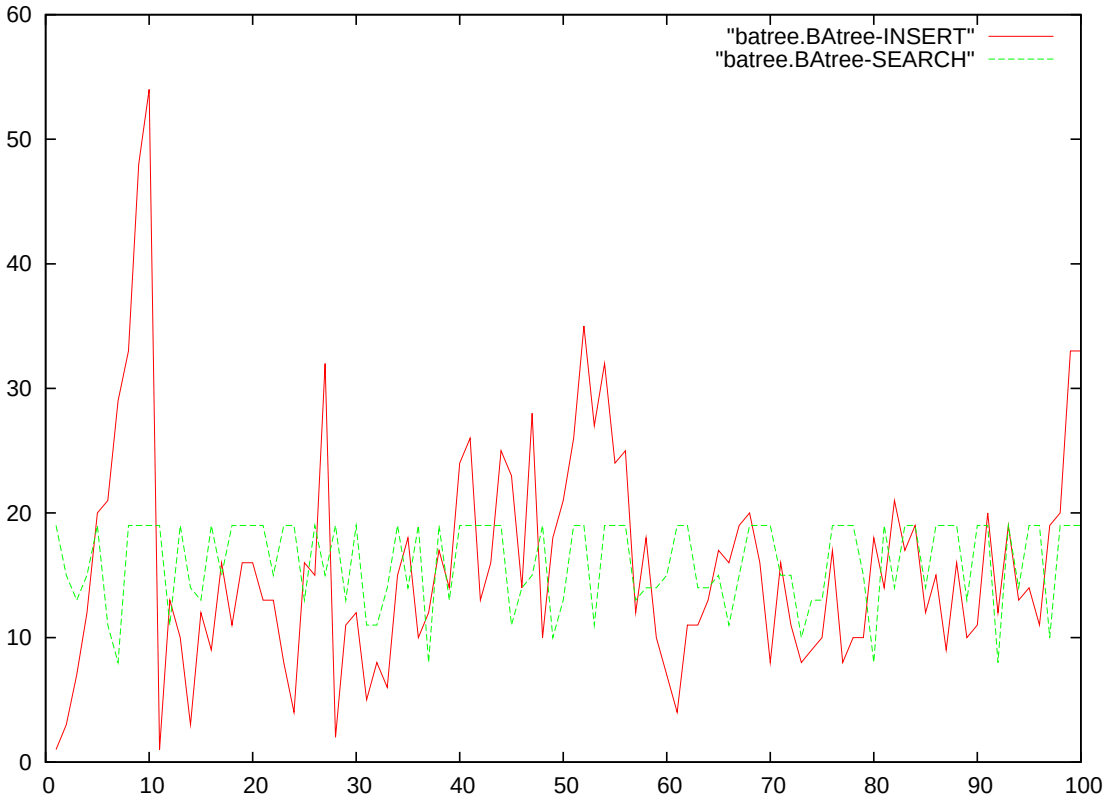
図 F.1: AR-tree の評価シミュレーション結果（UNIFORM）



(a) 対象データ

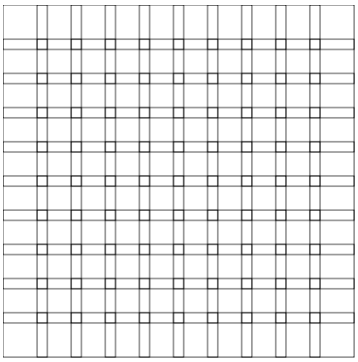


(b) 検索ツリーの深さ

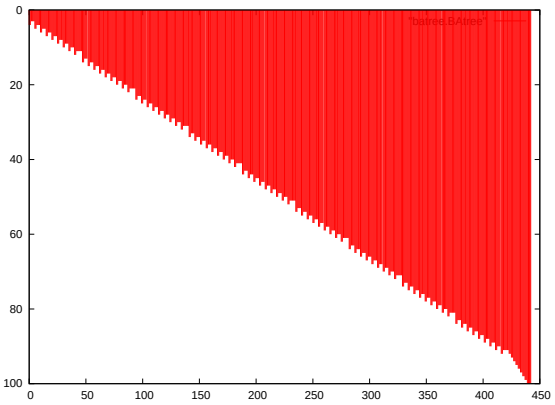


(c) 領域比較回数

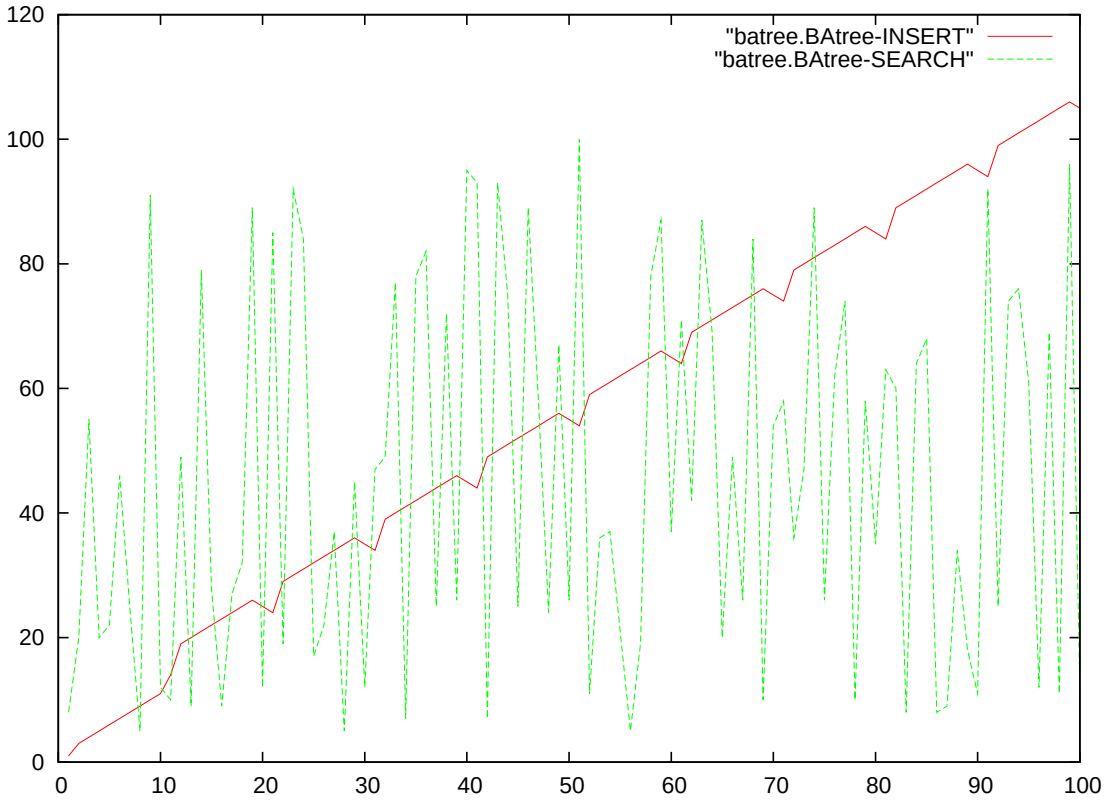
図 F.2: AR-tree の評価シミュレーション結果（MIX）



(a) 対象データ

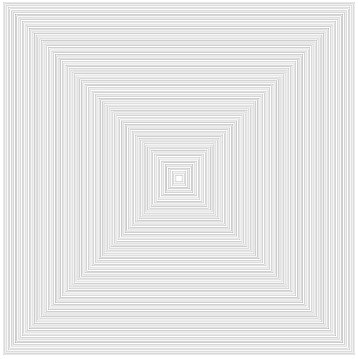


(b) 検索ツリーの深さ

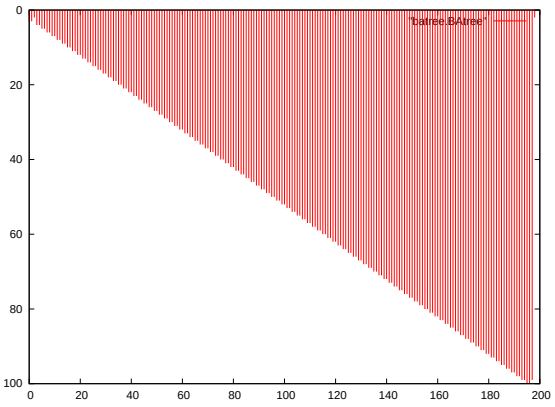


(c) 領域比較回数

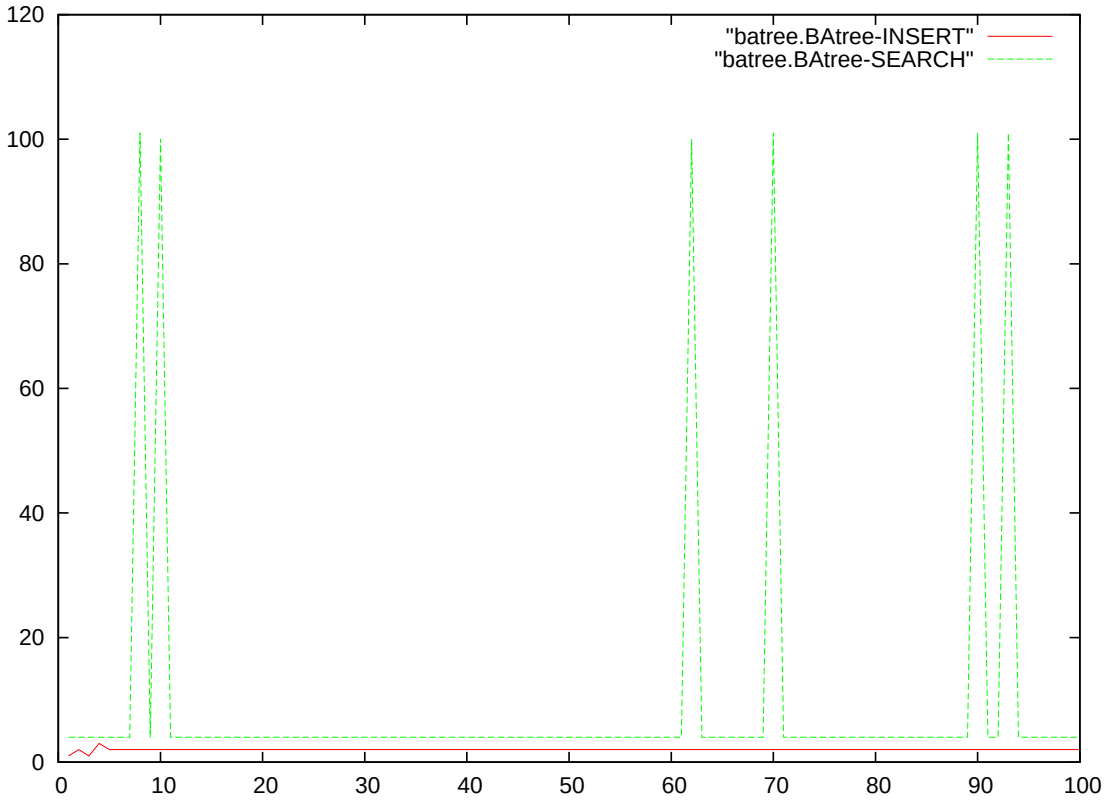
図 F.3: AR-tree の評価シミュレーション結果（PARCEL）



(a) 対象データ

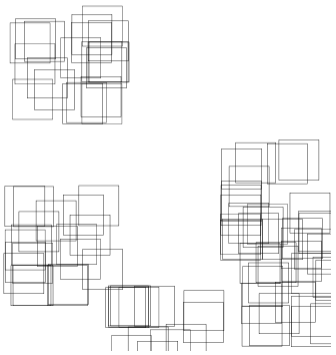


(b) 検索ツリーの深さ

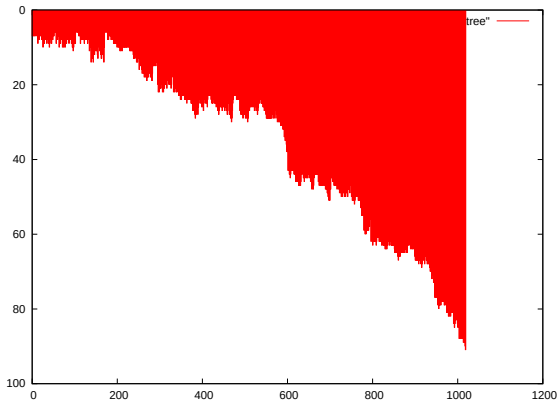


(c) 領域比較回数

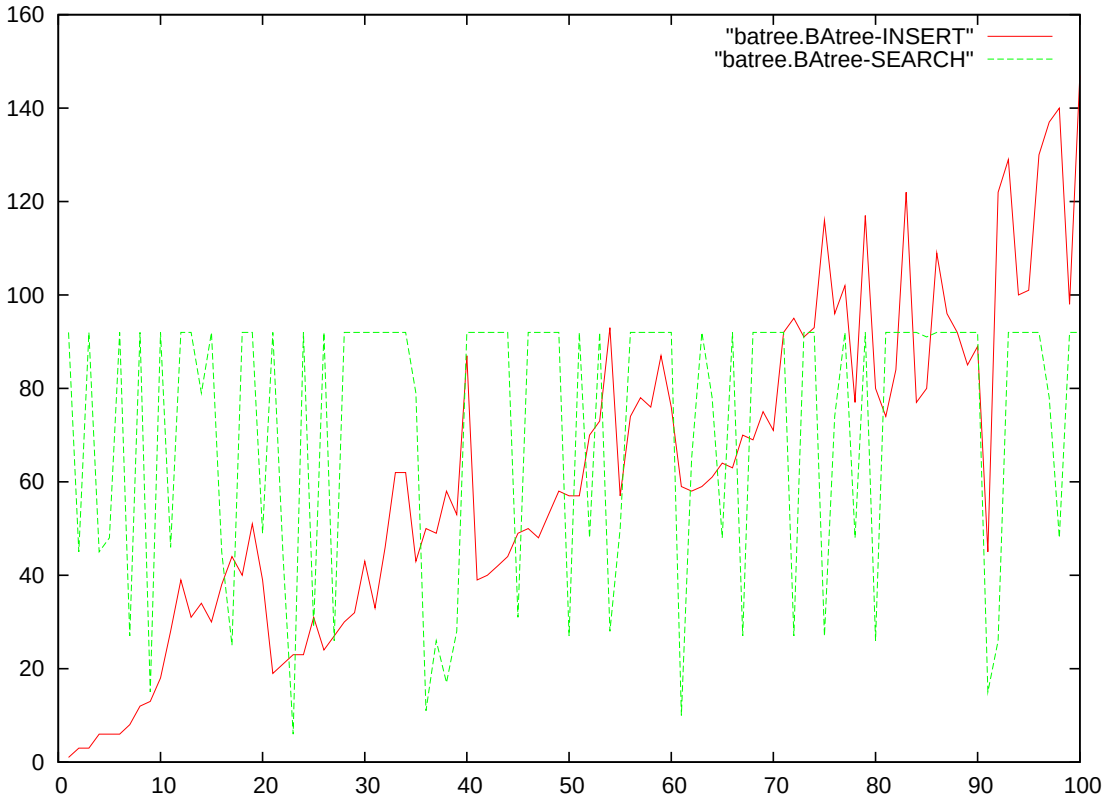
図 F.4: AR-tree の評価シミュレーション結果（CONCENT）



(a) 対象データ



(b) 検索ツリーの深さ



(c) 領域比較回数

図 F.5: AR-tree の評価シミュレーション結果（CLUSTER）

謝辞

本論文執筆にあたり，ご指導いただきました慶應義塾大学環境情報学部教授 村井純博士，並びに同学部教授 徳田英幸博士，同学部教授 中村修博士，同学部教授 武田圭史博士，同学部准教授 楠本博之博士，同学部准教授 高汐一紀博士，同学部准教授 植原啓介博士，同学部専任講師 重近範行博士，同学部専任講師 Rodney D. Van Meter 博士，同学部専任講師 中澤仁博士に感謝致します．

また，研究を進めるにあたり，ご指導とご助言をいただきました慶應義塾大学環境情報学部准教授 三次仁博士，並びに慶應義塾大学大学院政策・メディア研究科助教 鈴木茂哉氏，同学科助教 稲葉達也氏，同学科助教 中根雅文氏，同学科講師 羽田久一博士に感謝致します．特に，鈴木茂哉氏には本論文の執筆の際に，ご指導ご鞭撻いただきました．ご多忙な予定の中，親身にご指導いただけたことを深謝致します．氏なしには修士論文は完成し得ませんでした．

そして，慶應義塾大学徳田・村井合同研究室の研究生である空閑洋平氏，江村桂吾氏，佐藤泰介氏，田村哲朗氏，鈴木詩織氏，小澤みゆき氏，金仙麗氏，佐藤友紀氏，杉本健一氏，須山哲氏，富田千智氏，能島良和氏，山口修平氏，山田真弘氏，廣石達也氏，横石雄大氏，吉原洋樹氏，米村茂氏に感謝します．山田真弘氏，富田千智氏には図の作成を手伝っていただきました．重ねて感謝します．また，空閑洋平氏には本研究を進める際に多くの助言をいただきました．重ねて感謝します．

研究の場として OB/OG の諸先輩方が築き上げてきた研究団体の Auto-ID に感謝します．

最後に，私生活を支え，学びの場を与えてくれた父と母，そして，安らぎの場を築いてくれた祖母，姉，兄，友人への感謝をもって謝辞といたします．

2009 年 1 月 12 日
佐藤 龍

参考文献

[1] Live e! ～活きた地球の環境情報～. *http://www.live-e.org/*.

[2] Sensormap home. *http://atom.research.microsoft.com/sensewebv3/sensormap/*.

[3] デジタルサイネージとは【digital signage】 - 意味/解説/説明/定義 : it用語辞典. *http://e-words.jp/w/E38387E382B8E382BFE383ABE382B5E382A4E3838DE383BCE382B8.html*.

[4] 東京メトロ | ニュースリリース. *http://www.tokyometro.jp/news/2009/2009-34.html*.

[5] Futurize korea :: Gangnam media pole. *http://www.futurizekorea.com/293*.

[6] トレインチャンネル | jeki（株）ジェイアール東日本企画. *http://www.jeki.co.jp/transit/train/trainchannel/*.

[7] 三菱電機 交通システム：車両システム. *http://www.mitsubishielectric.co.jp/society/traffic/product/syaryou/s10.html*.

[8] 「ＴＯＫＹＯ　ＦＭ」福岡の電子看板実験に参加：経済ニュース：マネー・経済：Yomiuri online（読売新聞）. *http://www.yomiuri.co.jp/atmoney/news/20091218-OYT1T01097.htm*.

[9] 森川博之, 南正輝. 実空間指向ユビキタスネットワーク. 電子情報通信学会論文誌, Vol. 88, pp. 2137–2146, 2005.

[10] Mysql :: The world's most popular open source database. *http://www.mysql.com/*.

[11] Postgresql: The world's most advanced open source database. *http://www.postgresql.org/*.

[12] Sangikyo. *http://www.sangikyo.com/jp/products/mobile/n_dml.html*.

[13] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. pp. 47–57, 1984.

[14] Beckmann, H Kriegel, R Schneider, and B Seeger. The r*-tree: an efficient and robust access method for points and rectangles. *ACM SIGMOD Record*, Jan 1990.

[15] Mong Li Lee, Wynne Hsu, Christian S Jensen, Bin Cui, and Keng Lik Teo. Supporting frequent updates in r-trees: a bottom-up approach. pp. 608–619, 2003.

[16] R Finkel and J Bentley. Quad trees a data structure for retrieval on composite keys. *Acta informatica*, Jan 1974.

[17] Zhou Su, Masato Oguro, Jiro Katto, and Yasuhiko Yasuda. Selective update approach to maintain strong web consistency in dynamic content delivery. *IEICE Transactions*, Vol. 90-B, No. 10, pp. 2729–2737, 2007.

[18] Kiyohito YOSHIHARA, Hiroki HORIUCHI, Fumihide KOJIMA, Katsuyoshi SATO, and Masayuki FUJISE. Content delivery platform on its road-vehicle communication system based on radio over fiber(network). *IEICE transactions on information and systems*, Vol. 87, No. 2, pp. 426–435, Feb 2004.

[19] Takeshi YOSHIMURA, Yoshifumi YONEMOTO, Tomoyuki OHYA, Minoru ETOH, and Susie WEE. Content delivery network architecture for mobile streaming service enabled by smil modification(cdn architecture)(special issue on content delivery networks). *IEICE Transactions on Communications*, Vol. 86, No. 6, pp. 1778–1787, Jun 2003.

[20] Adobe – adobe air. *http://www.adobe.com/jp/*.

[21] Java profiler - .net profiler - the profilers for java and .net professionals. *http://www.yourkit.com/*.

[22] R-tree portal - home. *http://www.rtreeportal.org/index.php*.