

Work in Progress ～ P2P DB開発への道

その後 NICO は何をやっていたのか？

大阪市立大学大学院
創造都市研究科
藤田昭人

はじめに

- 第2回DHT勉強会での私の発表の続編です。
 - 別にシリーズ化を狙っているわけではない
- 前回の詳細は無印吉澤さんの議事録をご覧ください。
 - <http://muziyoshiz.jp/20061002.html>
- 前回のあらすじ
 - そもそも永続アーカイブを作るつもりだった
 - そのストレージシステムとして OceanStore を検討したが挫折
 - その代替りのシステムとして Chord/Dhash の実装を当たるがドツボにはまる
 - さらにその代替りのシステムとして i3 の実装にチャレンジするが・・・結局ダメダメ
 - 「さあ、この後どうなるのでしょうか？」・・・というところで終了
- 結論を次回に持ち越してしまったので・・・

構造化オーバーレイとは？

- ピア・ツー・ピア (P2P) ネットワーク技術の1つの研究領域
 - たくさんのピア(ノード)が集まってオーバーレイ・ネットワークを形成
 - 個々のピアはクライアントとしてもサーバーとしても動作する
 - **オーバーレイ・ネットワークの構造が決まっている(多くの場合、リング)**
- 大規模自律分散システムの基盤技術として活用できる
 - ノードをいくらでも接続できるから・・・
 - スケーラビリティ
 - 耐障害性
- 最近では P2P ファイル共有ソフトに活用されている

※ もう少し丁寧な説明は前回勉強会の首藤さんの講演資料を見てね

何故ネットワーク構造が決められるの？

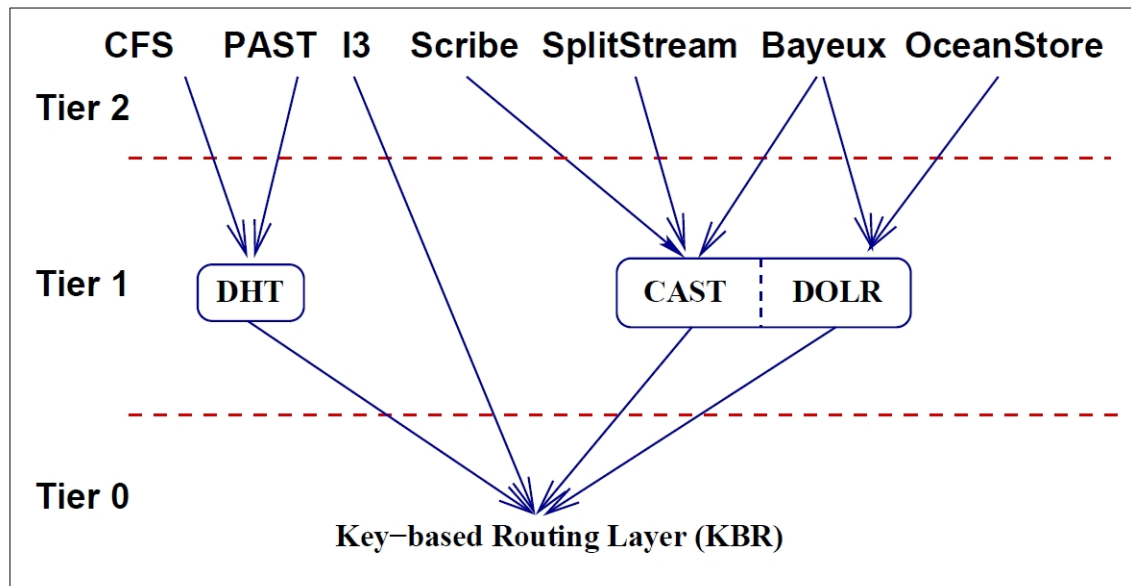
- **Key-Base Routing (KBR)** のおかげです。
 - KBRはキーを決めるとノードが決まるアルゴリズムです。
 - キーにはSHA-1などのハッシュ関数が使われます。(普通はね)
- オーバーレイ・ネットワークに参加するいずれかのノードにキーについて問い合わせると対応するノードを教えてください
 - 裏ではノードの間でメッセージのやり取りをして対応するノードを探し出します。(だからルーティング)
 - ノード間で交換するメッセージ数は最大で $O(\log N)$ になります。
- 対応するノードを高速に見つけ出すためにいろいろなプロトコル(アルゴリズム)が提案されています
 - Chord, Pastry, Tapestry, Kademlia …
 - 構造化オーバーレイの研究というところの問題を扱ってることが多い

DHTとは？

- ジェンドロテストステロンのことではありません
 - 「DHTが男性型脱毛症の原因物質です」
 - <http://ozaki-clinic.cocolog-nifty.com/blog/cat5745630/index.html>
- 分散ハッシュテーブルのことです
 - 構造化オーバーレイが提供する抽象化サービスの1つ
 - PUT/GET インターフェースを介して任意のキーとデータを格納できる
 - データをどのノードに格納するかはオーバーレイ構造で決定できる
 - 任意のノードがオーバーレイから離脱しても登録データは保存される
- 構造化オーバーレイ=DHTと思っている人も多いんだけど・・・

構造化オーバーレイ ≠ DHT

Towards a Common API for Structured Peer-to-Peer Overlays によると・・・
(<http://pdos.csail.mit.edu/papers/iptps:apis/main.pdf>)



DHT は Tier 1 Abstraction の1つです。

僕の抱えている問題

- Tier 1 Abstraction に定義されている抽象化サービスだけでは分散リレーショナル・データベースは実装できる？
 - データベース内のテーブルを複数のノードで共有したいのだけど・・・
 - もちろんDHTは使います。
でもDHTがサポートするのはデータの格納だけ。
 - テーブルへのアクセス競合は誰がどのように調停する？
- Tier 1 Abstraction の拡張が必要だ!!
 - 既存の Tier 1 Abstraction を拡張する？
 - 新たな Tier 1 Abstraction を定義する？
- Tier 1 Abstraction の改変が可能な構造化オーバーレイの実装が必要

Overlay Weaver

- ウタゴエの首藤さんが開発した構造化オーバーレイのツールキット
 - <http://overlayweaver.sourceforge.net/index-j.html>
- 最大の特徴は主要な KBR をサポートしていること
 - Chord、Kademlia、Koorde、Pastry、Tapestry
 - テーブル共有にどのプロトコル(アルゴリズム)が適しているか
とっかえひっかえ試してみることができる
- シミュレーションも簡単!!
 - メモリ1Gで4000ノードのシミュレーションが可能
→このノートPCでも動く
- わからないことがあれば直接聞ける！（それも日本語で！！）

Overlay Weaver

- Tier 1 Abstraction の改変について相談してみたら・・・
 - Overlay Weaver は KBR の抽象化に従っている。
 - DHT と CAST (のうちマルチキャスト) を実装
 - DHT 実装自体は非常に小さい
 - コア部分 (src/ow/dht/impl/DHTImpl.java) は642 ステップ (860 行)
 - KBR の API は独自
 - API は、src/ow/routing/RoutingService.java にある。
 - メソッドはとても少ない。本質的なのは・・・
 - join, leave, routeToRootNode, invokeCallbacksOnRoute
- 問題は Java の開発が僕にできるか？ということみたい・・・

Bamboo DHT

- Sean Rhea がメンテナンスしている(と思われる) DHT 実装
 - Churn 耐性を考慮した Pastry の拡張
<http://srhea.net/papers/bamboo-usenix.pdf>
- OpenDHT (<http://opendht.org/>) のエンジン
 - PlanetLab の上で稼動する公開 DHT サービス
 - Sun ONC RPC と XML-RPC の2種類のインターフェースで利用可能
<http://opendht.org/users-guide.html>
- プログラミングに関する Tutorial があつた (<http://bamboo-dht.org/tutorial.html>)
 - マルチキャストを扱う Tier 1 Abstraction の開発作業のレポートもあつた
<http://www.cl.cam.ac.uk/research/srg/netos/futuregrid/dischinger-report.pdf>

ReDiR

- DHT の上で KBR を動かす OpenDHT 用のクライアントライブラリ
 - <http://opendht.org/users-guide.html>
 - ハッシュ・キー空間を任意に定義することができる
 - ハッシュ・キーを指定して lookup を実行すると最も近傍のノード (successor) の情報 (IP address+port) が返ってくる
 - DHT に対しては put/get しか行わない
- 公開されているサンプル `redir++-0.3` を動かしてみた
 - `example` と `choredir_example` の2つのサンプル・プログラム
 - `example` は10台分の仮想ノードを生成するサンプル
 - `choredir_example` はなんと **Chord over Bamboo** だった

現時点での結論

- Overlay Weaver を使って Tier 1 Abstraction の追加を試みるつもり。
但し・・・何とか Overlay Weaver の外部で Tier 1 Abstraction を動かしたい
 - データベースのエンジンとして C で記述された SQLite を使っている
(Cで記述している部分とJavaで記述している部分をきれいに分けたい)
 - Overlay Weaver は未だ活発な開発作業が続いている
(moving target での開発は辛いよね)
 - Tier 1 Abstraction と KBR のインターフェースは研究的価値があるのでは？
 - 構造化オーバーレイ実装で一般的な Event-Driven Programming 関連の調査が必要？
- またまた to be continue 的まとめになっているような？

Third Trial - Experimental Network

- 筑波大学のひろきのだいちくんとゆうあんくんとで勢いで始めた構造化オーバーレイのための実験ネットワークです
- キッカケは・・・前回の勉強会のあとの懇親会
 - やっぱゲリラ的にやらないと実験なんかできないジャン!!
- 双方「論文を書く」という約束を担保にマシンを調達
- Noritsuna さんの協力もあって現在14ノード確保、11ノード稼動中
- 標準DHTは Overlay Weaver です
 - 首藤さんがシミュレーションを流して「4000ノードで失敗」とのこと
 - 1000ノードまではOK・・・だろう

Third Trial - Experimental Network

- 当面の目標は30～50ノードを集めること
 - これだけあればそれなりのベンチマークができる
 - 仮想ノード機能を活用できれば数千～数万規模のオーバーレイも
- 敢えて独自に実験ネットワークを作る意義は？
 - 日本国内での構造化オーバーレイ研究の活性化に繋がる
(企業がPlanetLabに参加するのはコスト的に厳しい)
 - 構造化オーバーレイのビジネス化のデモンストレーション環境
(政府系予算を獲得する上では国内で動いていること重要?)
- みなさん、是非ご協力を!!