第5章 数十万の経路探索に耐えるアルゴリズム「パトリシア・トライ」を実装する

ステップ④…ルーティング&フォワーディング

柚山 大哉

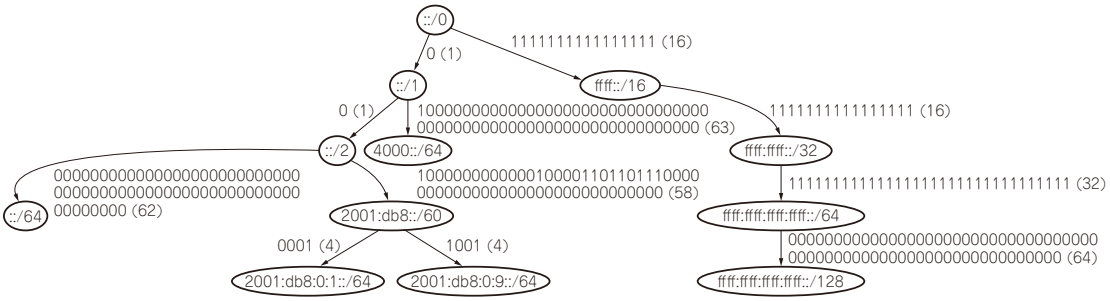

図1　数十万の経路情報を保持した探索に耐えるアルゴリズム「パトリシア・トライ」の木構造

本章では，IPアドレスに応じてどこのネットワークからパケットを送信するかを決定するIPルーティングとフォワーディングについて解説し，ルータ・プログラムにコードを実装します．

IPv6におけるフォワーディングの基礎知識

● 経路を決めるフォワーディング・テーブル

宛先IPアドレスをキーとして，そのパケットを次にどこへ送信すればよいかを決定するためにはテーブルが必要です．これをフォワーディング・テーブルと呼びます．経路が数個しかない場合は，配列で経路情報を持っておいて，毎回線形探索で1つずつマッチするか調べる方法が使えます．

しかし，インターネットのバックボーンで使用されているルータでは，数十万の経路情報があり，線形探索を使うのは現実的ではありません．そのようなユースケースにも耐えられるように，今回は実際のソフトウェア・ルータでも使用されている高速なアルゴリズムを実装します．

● 高速アルゴリズム「パトリシア・トライ」

IPv6はアドレス長が非常に長いので，IPフォワーディング・テーブルを作るときは，データ構造に注意する必要があります．

必ず1ビットずつ分岐するバイナリ・トライは実装が比較的簡単ですが，IPv6のアドレスのルックアップの際に最大128回ポインタをたどる必要があります(/128に対する経路の場合)．それを解決するために，よりポインタをたどる回数を減らすアプローチを考えてみます．

パトリシア・トライは，二分木の一種で，子が1つだけのノードを圧縮することで木の長さを削減したデータ構造です．1つのノードで複数のビットを保持できるので，ポインタをたどる回数を削減でき，高速化を図れます．

幾つかのエントリを登録した木構造は，**図1**のようになります．

リスト1　ノードの構造体patricia_node

```
struct patricia_node {
    patricia_node *left, *right, *parent;
    in6_addr address; // IPv6アドレス
    int bits_len; // このノードで比較するビットの位置
    int is_prefix; // このノードがプレフィックスを表すかどうか
    void* data;
};
```

左右のノード

親ノード

IPルーティング&フォワーディングの実装

● ノードに関する構造体・関数

▶ノードの構造体

ノードの構造体を**リスト1**に示します．