大量のコネクションをさばく サーバーの作り方

山本和彦

(株)インターネットイニシアティブ

kazu@iij.ad.jp

未だに

「Haskellって実用的なの？」

と訊く人がいますが…

Simon PEYTON JONES

によると

Haskell は
世界で最も素晴らしい
命令型言語だ

知ってました？

素晴らしい命令型言語である証拠に

今日は

大量のコネクションをさばくサーバーを

Haskell で作るお話をします

今日の目標

1万本のコネクションをさばく

17行で作るエコーサーバー

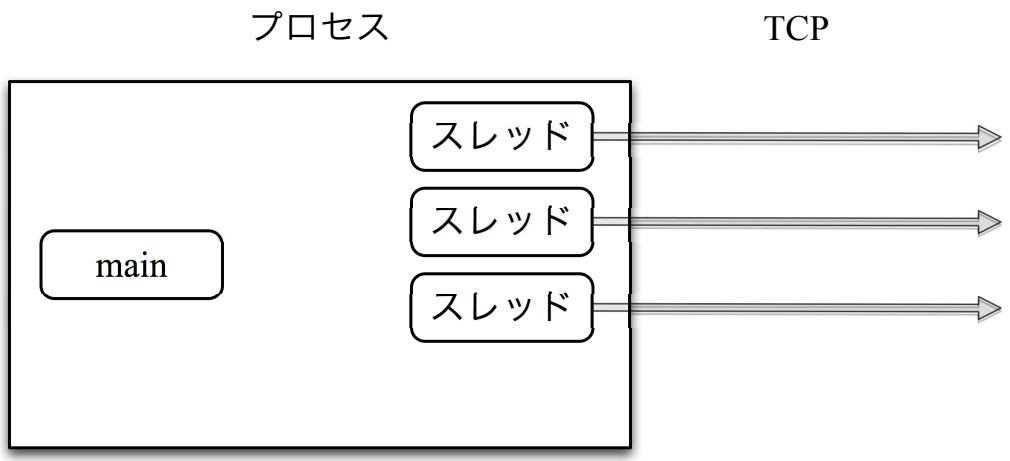
```
import Control.Concurrent
import Network
import System.IO

main :: IO ()
main = do
    sock <- listenOn (Service "2000")
    dispatch sock

dispatch :: Socket -> IO ()
dispatch sock = do
    (hdl,_,_) <- accept sock
    forkIO (server hdl) -- スレッドを作成
    dispatch sock

server :: Handle -> IO ()
server hdl = do
    cs <- hGetLine hdl
    hPutStrLn hdl cs
    hClose hdl
```

夢のような
スレッドプログラミング



しかし

kqueue() や epoll() の時代に

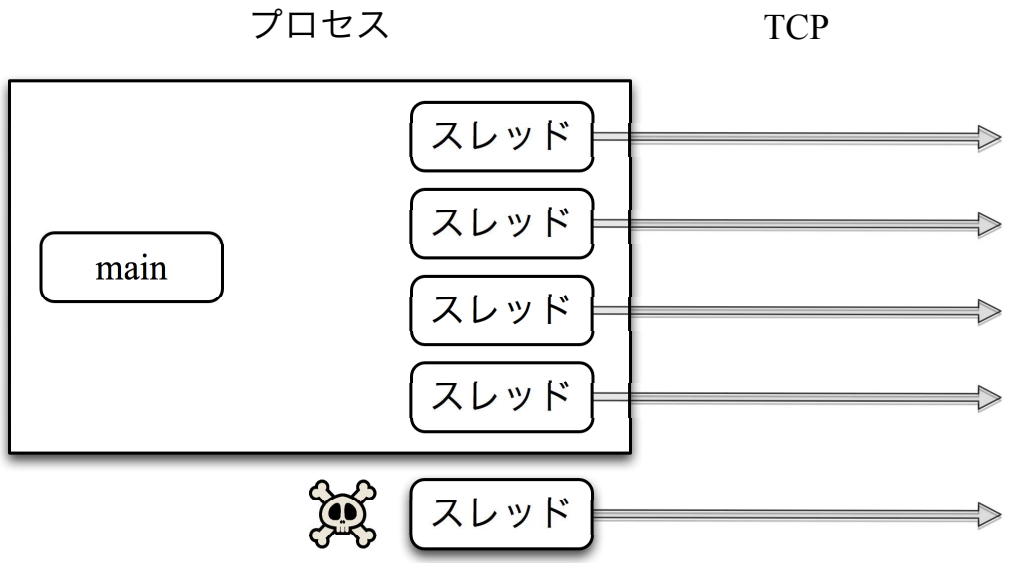
Haskell スレッドは

select()

で実装されている

select() なので

コネクションを1024以上
受け付けるとエラーになる



なんとかしないと！

大量のコネクションをさばくための 3ステップ

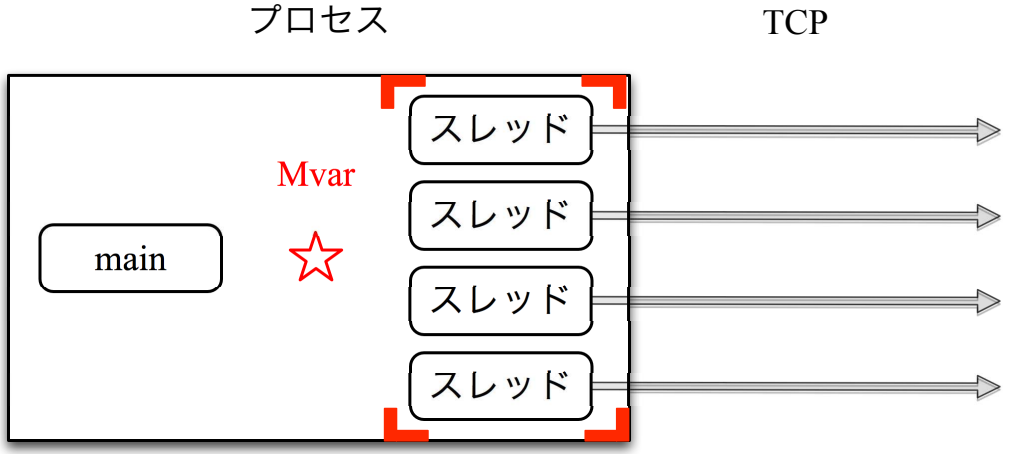
1) コネクションの数を制限

2) 黙り込んだ相手のコネクションを切る

3) プロセスを prefork する

1) コネクションの数を制限

- スレッド間の共有変数 Mvar



- main スレッドは子の生成時に Mvar の値を増やす
- 子スレッドは終了時に Mvar の値を減らす

```
forkIO (server hdl) ‘finally‘ (decMvar mvar)
```

- main スレッドは上限に達したら一定期間寝る

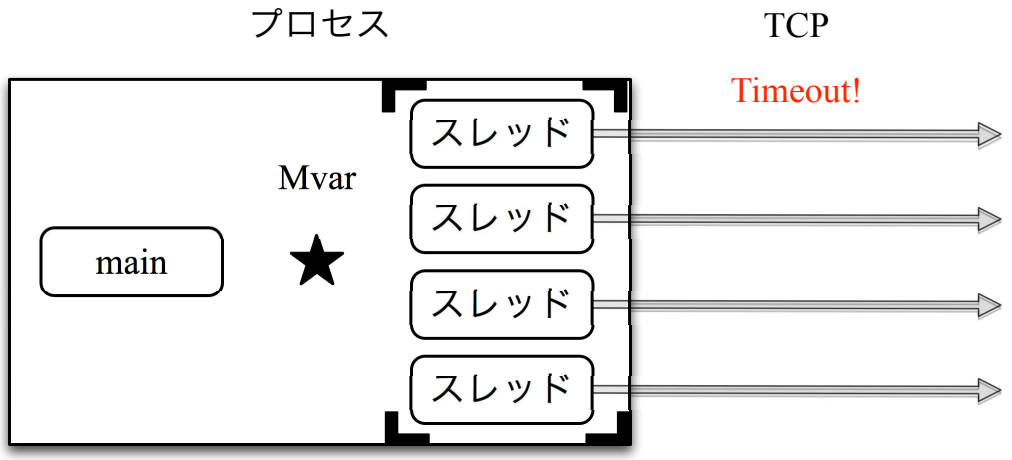
上限を 1000 とすると

1000本のコネクションを
安心してさばけるようになった

2) 黙り込んだ相手のコネクションを切る

- 相手が黙ると読み込みはブロックする
  - `hGetLine :: Handle -> IO String`
- Timeout コンビネーター

  `timeout :: Int -> IO a -> IO (Maybe a)`
  - 読み込めたら → `Just` 文字列
  - タイムアウトしたら → `Nothing`



いじわるなクライアントがいても

1000本のまともなコネクションを
さばけるようになった

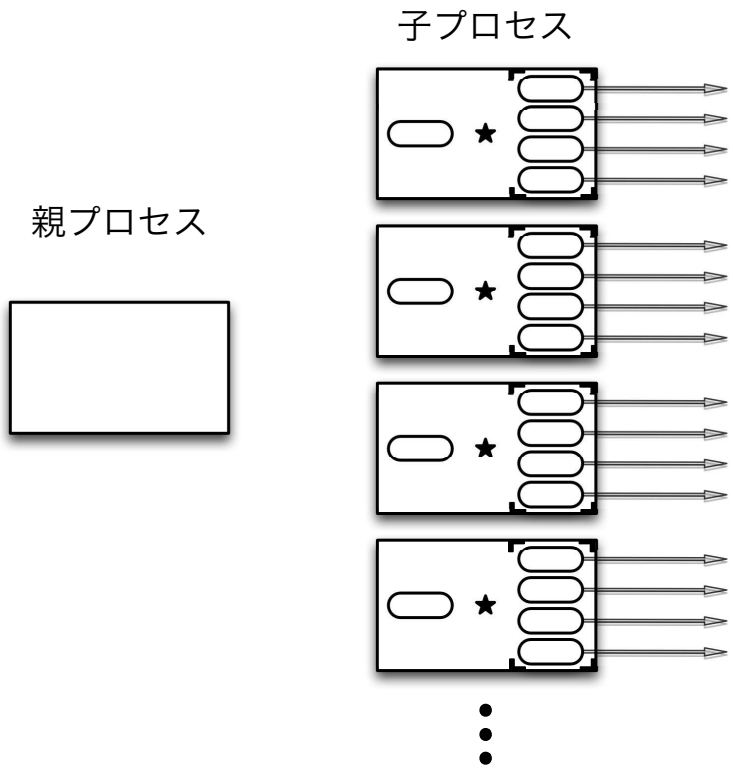
3) プロセスを prefork する

- 普通は accept() の後に fork() するが…
- accept の前に forkProcess する

```
main :: IO ()
main = do
    sock <- listenOn (Service "2000")
    replicateM_ 10 $ forkProcess (dispatch sock)
    mySleep -- threadDelay をループ

dispatch :: Socket -> IO ()
dispatch sock = do
    (hdl,_,_) <- accept sock
    forkIO (server hdl) -- スレッドを作成
    dispatch sock
```

プロセスを10個 prefork すると



10×1000本のコネクションを
さばけるようになった

目標達成！

最後に

Web サーバーの

デモとはいえないデモ