Industrial Functional Programming [1]

Melinda Tóth, István Bozó

Dept. Programming Languages and Compilers
Eötvös Loránd University, Budapest, Hungary

[1] Supported by TÁMOP-4.1.2.A/1-11/1-2011-0052

Contents

Functional Programming

- The topmost level is a set of modules
- The module is a set of declaration (type, class, function)
- Initial statement
- Evaluation
- Based on mathematical model (Lambda Calculus)
- Turing complete

Functional Programming Languages

- Lisp,
- Haskell, Clean,
- Scheme, SML,
- **Erlang**, Scala,
- F#, OCaml,
- Miranda, Closure,
- Agda, Epigram,
- etc.

Factorial

How can we calculate the factorial of a number?

Factorial

How can we calculate the factorial of a number?

- 0! = 1
- N! = N*(N-1)!

Factorial Calculation in Erlang

```
fact(0) -> 1;
```

$$0! = 1$$

```
fact(N) -> N * fact(N-1).
```

$$N! = N * (N-1)!$$

Properties

- Referential transparency
- (Static typing)
- Higher-order functions
- (Currying)
- Recursion
- Strict(/lazy) evaluation
- List comprehensions
- Pattern matching
- (“Offset rule”)
- IO model

Properties

- **(Static typing)**

```
-spec(fact(pos_int())->pos_int())
```

```
fact(-3.5)
```

Properties

- **Higher-order functions**
- Mathematical example: differential calculus

```
m_fun(Fun, A, B) -> Fun(A, B).
```

```
m_fun(fun add/2, A, B)
m_fun(fun mul/2, A, B)
```

Properties

- **(Currying)**

```
add(A, B)  = A + B.
add(1)     = fun(B) -> 1 + B end = inc(B).
inc(3)     = 4.
add(1)(3)  = 4.
```

Properties

- **Recursion**

```
fact(0) -> 1;
fact(N) -> N * fact(N-1).
```

Properties

- **Strict(/lazy) evaluation**
- Calculate the first three natural number!

```
take([1..])
```

Properties

- **List Comprehensions (Zermelo-Frankel set-expressions)**
- Calculate the square of the first two hundred even natural number!

```
{x^2  |  x  in N,    x < 200,   2 | x}

[x*x  |  x <- [1..], x < 200,   x mod 2 == 0]

[X*X  || X <- lists:seq(1,200), X rem 2 == 0]
```

Properties

- **Pattern Matching**

```
[Head | Tail ] = [1,2,3,4,5,6]

 Head = 1
 Tail = [2,3,4,5,6]

{X, Y, Z}   = {1,2,3}

 X = 1
 Y = 2
 Z = 3
```

Properties

- **(Offset rule)**

```
fact 0 = 1
"  "fact x = x * fact x-1

fact 0 = 1
fact x = x * fact x-1
```

Properties

- **Referential Transparency**

```
fact(6) == 720
...
fact(6) == 720
```

Properties

- **Purity/Side-effects**

History

- 1982 - 1986 – Experiments with different programming languages
- 1987 – First experiments with Erlang
- 1988 - 1990 – Experiences with Erlang in telecom world
- 1993 – Distributed programming / First Erlang book (The BOOK)
- 1996 – OTP R1
- 1998 – Released as Open Source
- 2005 – R11 multicore

Erlang – Properties

- Declarative – Functional programming language, high level of abstraction
- Dynamically typed
- Concurrency – explicit concurrency, LWP
- Soft real-time characteristics
- Robustness – supervision trees
- Distribution – transparent, explicit, network
- Openness, external interfaces – “ports”
- Portability – Unix, Win., ... , heterogeneous network
- SMP Support – multicore
- “Hot code loading”

Erlang – Ericsson Language

- Erlang, Agner Krarup (1878-1929)
- Danish mathematician
- Erlang formula
- erlang – unit of load on telephone circuits

When To Use Erlang?

- Complex, continuously operating, scalable, maintainable, distributed
- Rapid and efficient development
- Fault-tolerant (software, hardware) systems
- Hot-code loading

Who Uses Erlang?

- Ericsson – telecommunication (AXD301 ATM switch), simulation, testing, 3G, GPRS
- Amazon – Simple DB (DBMS)
- Yahoo – Online bookmarks service
- Facebook – chat server
- T-Mobile – SMS gateway
- Motorola – call processing
- MochiWeb – http server
- CouchDb – document database server (multicore, multiserver clusters)
- YAWS – Yet Another Web Server
- Wings3D – 3D modeling
- and many other...

Literature and materials to use

- `erlang.org`
- Erlang Programming: A Concurrent Approach to Software Development by Francesco Cesarini, Simon Thompson. O'Reilly Media
- Programming Erlang: Software for a Concurrent World by Joe Armstrong. The Pragmatic Bookshelf
- Learn You Some Erlang for Great Good! A Beginner's Guide by Fred Hebert. No Starch Press.
- `erlang-factory.org`

On the Next Lecture ...

- The Erlang VM
- Erlang Terms
- Variables
- Pattern Matching