Зміст

1. [1. Загальні відомості про Центр суперкомп’ютерних обчислень НТУУ «КПІ» 2](#_Toc393967881)
2. [**Вступ** 4](#_Toc393967882)

[2.1 Типи даних 4](#_Toc393967883)

[2.1.1 Числа 5](#_Toc393967884)

[2.1.2 Атоми 5](#_Toc393967885)

[2.1.3 Бітові рядки і бінарні дані 6](#_Toc393967886)

[2.1.4 Кортеж 6](#_Toc393967887)

[2.1.5 Список 7](#_Toc393967888)

[2.1.6 Рядок 8](#_Toc393967889)

[2.1.7 Логічні значення 8](#_Toc393967890)

[2.1.8 Функціональний об'єкт (Fun) 8](#_Toc393967891)

[2.1.9 Запис (record) 9](#_Toc393967892)

[2.2 Операції 9](#_Toc393967893)

[2.2.1 Зіставлення зі зразком 9](#_Toc393967894)

[2.2.2 Функції 9](#_Toc393967895)

1. [Опис роботи 11](#_Toc393967896)
2. [Висновки 12](#_Toc393967897)

[Список використаних ресурсів 13](#_Toc393967898)

# Загальні відомості про Центр суперкомп’ютерних обчислень НТУУ «КПІ»

Центр суперкомп'ютерних обчислень (ЦСВ) був створений з метою розвитку інформаційної інфраструктури освіти і науки, реалізації положень Указу Президента України від 20 жовтня 2005 року № 1497.2005 «Про першочергові завдання щодо впровадження новітніх інформаційних технологій», завдань Державної цільової програми «Інформаційні та комунікаційні технології в освіті і науці на 2006-2010 роки»та на виконання відповідного розпорядження Кабінету Міністрів України № 301-р від 31 травня 2006.

На технічному майданчику ЦСВ розташовується обладнання Світового центру даних (СЦД) з геоінформатики та сталого розвитку. СЦД має окремий зовнішній канал доступу до мережі інтернет. Функціонування СЦД забезпечується службовими серверами, які об'єднані в окрему підмережу, і серверами зберігання на основі ПЗ IpStor (6,5 Тб у відмовостійкій системі).

Технічний майданчик обладнано системою регулювання мікроклімату на основі обладнання InRow фірми APC з водним охолодженням, а також системою безпеки живлення APC Symmetra, що захищає від перепадів напруги і надає можливість зниження споживаної кластером потужності за рахунок коректного відключення обчислювальних вузлів в разі аварії або повної відсутності живлення.

За свою історію кластерний суперкомп'ютер НТУУ «КПІ» пройшов два оновлення, кожне з яких збільшувало його обчислювальну потужність: починаючи з 44 вузлів, мали пікову продуктивність близько 2 Тфлопс, ​​основний кластер зараз складається з 112 вузлів з 624 процесорами, пікова продуктивність яких становить 7 Тфлопс (linpack 5,7 Тфлопс).

Поточна конфігурація кластера НТУУ «КПІ» має у своєму складі 2 системи.

Перша забезпечує основу потужності і працює під управлінням ОС Linux.

На базі другої функціонує кластер під управлінням ОС MS Windows HPC Edition, а також навчальний кластер, який служить для проведення навчальних курсів і забезпечення лабораторних робіт.

Внутрішній інтерконект в обох системах реалізований на базі окремих комутаторів технології Infiniband. Службова мережа, яка об'єднує кластер, сервісні сервери та обслуговуючу лабораторію, побудована за технологією Gigabit Ethernet.

Зовнішній канал зв'язку забезпечується оптоволоконної лінією КПІ-Телеком, по якій кластер має доступ до ресурсів мережі URAN.

1. **Вступ**

Erlang - функціональна мова програмування зі строгою динамічною типізацією, призначена для створення розподілених обчислювальних систем. Розроблена та підтримується компанією Ericsson. Мова включає в себе засоби породження паралельних полегшених процесів та їх взаємодії через обмін асинхронними повідомленнями відповідно до моделі акторів.

Erlang був цілеспрямовано розроблений для застосування в розподілених, відмовостійких, паралельних системах реального часу, для яких крім коштів самої мови є стандартна бібліотека модулів і бібліотека шаблонних рішень (так званих поводжень) - фреймворк OTP (англ. Open Telecom Platform). Програма на Erlang транслюється в байт-код, виконуваний віртуальними машинами, що знаходяться на різних вузлах розподіленої обчислювальної мережі. Erlang-системи підтримують гарячу заміну коду, що дозволяє експлуатувати обладнання безупинно.

Свій синтаксис і деякі концепції Erlang успадкував від мови логічного програмування Пролог. Мова підтримує багато типів даних, умовні конструкції, зіставлення зі зразком, обробку винятків, спискові включення і вирази бітових рядків, функції (анонімні функції, функції вищого порядку, рекурсивні визначення функцій, оптимізацію хвостовій рекурсії), модулі, прийом і відправку повідомлень між процесами. Препроцесор підтримує роботу з макросами і включення заголовних файлів.

Популярність Erlang почала зростати у зв'язку з розширенням його області застосування (телекомунікаційні системи) на високонавантажених паралельні розподілені системи, обслуговуючі мільйони користувачів WWW, такі як чати, системи управління вмістом, веб-сервери та розподілені, що вимагають масштабування бази даних. Erlang застосовується в декількох NoSQL-базах даних високої доступності.

## Типи даних

Типізація в Erlang є строгою і динамічною. Динамічна типізація була обрана для мови Erlang тому, що перші розробники були більше з нею знайомі. На думку Джо Армстронга, статична типізація зажадала б дуже великих трудовитрат, зокрема, реалізувати систему гарячої дозавантаження коду було б украй важко. Така типізація, при якій можливі помилки типів виявляються тільки під час виконання, тим не менш, не завадила створювати системи з дуже високим рівнем доступності. Дані в Erlang є незмінними: операції не переписують старі значення, що знаходяться в пам'яті.

### 2.1.1 Числа

У Erlang є два типи числових літералів: цілі і з плаваючою комою, наприклад: 4.5e-20, 125. Крім звичайної нотації, числа можна задавати через символ ASCII (наприклад, $ B означає 66) або разом із зазначенням системи числення з основою від 2 до 36 (у старих версіях - до 16), наприклад: 16 # 3f, 2 # 1010. У Erlang застосовуються цілі числа довільної точності і дійсні числа подвійної точності (64 біта), в стандарті IEEE 754-1985.

Приклад обчислень в інтерактивній оболонці:

Erlang R15B01 (erts-5.9.1) [source] [64-bit] [smp: 4:4] [async-threads: 0] [kernel-poll: false]

 Eshell V5.9.1 (abort with ^ G)

1> 123/23 + 12 \* (2 +3).

65.34782608695652

2> math: cos (math: pi ()).

-1.0

3> random: uniform (10).

5

### 2.1.2 Атоми

Атом - константа з ім'ям, яка має бути укладена в одинарні лапки, якщо не починається з малої букви або містить знаки, крім букв, цифр, підкреслення, крапки і символу @. Поняття атома запозичене з Прологу і його можна вважати аналогом перерахувань (enum) в інших мовах програмування (без необхідності попередньої декларації). Атоми використовуються майже виключно в порівняннях, що мають у Erlang дуже ефективну реалізацію. Крім того, деякі атоми мають певний сенс у возращаться значеннях і описі винятків. До них відносяться: error, ignore, noreply, ok, reply, stop, undefined.

### 2.1.3 Бітові рядки і бінарні дані

Бітова рядок використовується для зберігання в пам'яті нетипізований даних. Рядки, що складаються з цілого кількості октетів, називаються бінарними (або двійковими) даними (англ. binaries). Синтаксис опису бітової рядки досить гнучкий, тому що описує значення бітів окремих діапазонів і може бути забезпечений модифікатором типу. Кілька прикладів в інтерактивній командній оболонці:

1> << 23,89,120 >>.

<< 23,89,120 >>

2> << "ABC" >>.

<< 65,66,67 >>

3> << 10,17,42:16 >>.

<< 1,17,0,42 >>

4> << $ a, $ b, $ c >>.

<< "abc" >>

5> << 1024/utf8 >>.

<< 208,128 >>

Вирази бітових рядків (англ. bitstring comprehension) аналогічні списковую включениям, але працюють над бітовими рядками :

1> <<<< bnot (X): 1 >> | | << X: 1 >> <= << 2 # 111011:6 >>>>.

<< 4:6 >>

### 2.1.4 Кортеж

Кортеж (англ. tuple) - складовий тип даних з фіксованою кількістю елементів. При доступі до елементів кортежу за допомогою вбудованих функцій нумерація елементів починається з одиниці, а не з нуля. Перший елемент кортежу прийнято використовувати для вказівки ролі кортежу в програмі. Якщо перший елемент - атом, його називають тегом (англ. tag - «мітка»). У Erlang прийнято будувати різні типи даних на основі кортежів з тегами, що полегшує налагодження програми і вважається хорошим стилем програмування.

Для роботи з кортежами є кілька вбудованих функцій, наприклад :

1> tuple\_size ({a, 1, "777"}).

3

2> element (1, {b, 2, 3, 4}).

b

3> setelement (1, {c, 5}, d).

{d, 5}

### 2.1.5 Список

Список (англ. list) - складової тип даних, що містить змінне число елементів. Для маніпуляції зі списками можна застосовувати функції модуля lists стандартної бібліотеки. Формально список визначається як має голову (англ. head) і хвіст (англ. tail), що виражається синтаксично у вигляді [HEAD | TAIL], де хвіст зазвичай є списком (можливо, порожнім). Порожній список позначається [].

Списки можна записувати і більш звичним способом. Наступні записи еквівалентні:

1> [a | [b | [c | []]]].

[a, b, c]

Для роботи зі списками можна застосовувати спискові включення (генератори списків), наприклад:

1> [X / 2 | | X <- [1,2,3,4]].

[0.5,1.0,1.5,2.0]

### 2.1.6 Рядок

У Erlang немає самостійного типу для рядків: внутрішньо рядки представляються списками. Синтаксично рядок можна задати лапками. Так, "Привіт!" рівносильний (в підходящої кодуванні) списком [1055,1088,1080,1074,1077,1090,33]. Erlang підтримує Unicode як в рядку, так і в запису окремого знака (через $).

Атоми і рядки зовні досить схожі, але мають зовсім різні реалізації. Тоді як атоми можна тільки порівнювати, рядки підтримують багато інших операцій, для них є безліч функцій в модулях lists і string. Рядок може виконувати функції атома, але пам'ять, займана рядком пропорційна її довжині, тоді як атоми зберігаються в системній таблиці і на кожне використання атома в програмі доводиться лише пара байтів, незалежно від довжини атома. Порівняння двох атомів - це порівняння двох внутрішніх ідентифікаторів, що виконується за одну операцію, тоді як порівняння рядків припускає поелементний прохід елементів рядків.

### 2.1.7 Логічні значення

Для значень істина і брехня в Erlang застосовуються атоми true (істина) і false (неправда), які і використовуються операціями порівняння, логічними операціями, вбудованими функціями. Приклад:

1> 2 <3.

true

2> is\_boolean (125).

false

### 2.1.8 Функціональний об'єкт (Fun)

Fun-вираз дозволяє створити анонімну функцію, наприклад, для передачі як параметр іншим функцій. За допомогою fun можна також отримати функціональний об'єкт для функції з модуля. Приклади:

1> lists: map (fun (X) -> X + 1 end, [1, 2, 3]).

[2,3,4]

2> Belongs = fun lists: member / 2.

# Fun <lists.member.2>

3> Belongs (a, [a, b]).

true

### 2.1.9 Запис (record)

Щоб позначати окремі елементи кортежів і уникнути помилок при написанні програми, в Erlang був внесений синтаксис записів (англ. record). Для роботи з записами, необхідно на початку дати опис записи директивою-record, наприклад, для запису user опис може бути наступним :

-record (user, {login = "anon", password, nickname}).

* 1. **Операції**
     1. **Зіставлення зі зразком**

Зіставлення із зразком використовується в Erlang для присвоювання (у тому числі, при роботі з параметрами функцій), управління потоком виконання програми, вилучення значень складових типів, вибору повідомлення з черги. У лівій частині порівняння (або в заголовку функції) можуть перебувати пов'язані (уже мають значення) і незв'язані (одержують значення) змінні, а також літерали (атоми, числа, рядка). В результаті виконання порівняння може виявитися успішним (у цьому випадку змінні зв'язуються зі значеннями) і неуспішним - змінні залишаються непов'язаними. У зразку можуть бути змінні, значення яких для зразка байдуже: їх імена записуються починаються з підкреслення. Змінна з ім'ям \_ (підкреслення) зіставляється з будь-яким значенням, але при цьому не відбувається зв'язування. Таку зміну можна застосовувати багато разів.

* + 1. **Функції**

Програми на Erlang складаються з функцій, які викликають один одного. Кількість параметрів функції називається арністю. При виклику функції заголовні частини опису функції зіставляються зі зразком. У разі збігу параметрів виклику, формальні параметри зв'язуються з фактичними і виповнюється відповідна частина тіла функції. Запис варіанту обчислення функції для деякого зразка може називається Клозе від англ. clause, а визначення функції - це набір з одного або більше Клозе.

Для уточнення зіставлення з зразків у функціях можна використовувати охоронні вирази, які слідують після ключового слова when. У прикладі нижче визначена функція обчислення знака числа, яка розраховується залежно від порівняння параметра з нулем:

sign (X) when X> 0 -> 1;

sign (X) when X == 0 -> 0;

sign (X) when X <0 -> -1.

Клози Erlang перебирає в тому порядку, в якому вони записані, поки не буде знайдений відповідний заголовок. В охоронних виразах можна використовувати тільки обмежений набір вбудованих функцій, так як ці функції не повинні мати побічних ефектів.

1. **Опис роботи**

На мові програмування Erlang був написаний модуль для сервера Ejabberd - вільного (GNU GPL), розподіленого і стійкого до відмов XMPP-сервера, написаного в основному на Erlang, який працює під Unix-подібними ОС (GNU / Linux, FreeBSD і NetBSD).

Модуль призначений для обробки певного виду запитів користувачів, а саме перевірки приналежності їхнього списку контактів до працюючого на розподіленій обчислювальній системі ejabberd-сервера.

Модуль реалізує поведінку (інтерфейс) gen\_mod та gen\_server, тобто являється одночасно і модулем для ejabberd-сервера і незалежним потоком, супервізором якого також є цей сервер. Це дає можливість одночасно краще інтегрувати його в існуючу систему і підтримати відмовостійкість основного ejabberd-сервера у разі відмови розробленого модуля.

При запуску модуля він реєструє роутигнг (перенаправлення) пакетів, які прийшли на певну адресу (вказану у конфігураційному файлі) від користувачів на власну обробку. Обробка включає: логування (журнальний запис) запитів у файл, парсинг пакета, виділення тіла (інформаційної частини) запиту, парсинг тіла (розбиття його) на ідентифікатори (jid) контактів, перевірка в базі даних сервера на їх наявність, збирання та відправка пакета результату.

Написаний модуль працює як складова частина ejabberd-сервера, його можна дуже просто увімкнути або вимкнути, не зупиняючи при цьому роботи сервера.

1. **Висновки**

Мова програмування Erlang є дуже зручною для розробки розподілених обчислювальних систем та написання відмовостійких програм, які здатні обробляти дуже велику кількість запитів користувачів, вона була ціле направлено розроблена для цього. Програма на Erlang транслюється в байт-код, виконуваний віртуальними машинами, що знаходяться на різних вузлах розподіленої обчислювальної мережі.

Висока відмовостійкість криється у застосуванні ізольованих один від одного полегшених процесів, пов'язаних лише механізмом обміну повідомленнями і сигналами виходу. Відмінною особливістю мови є застосування полегшених процесів відповідно з моделлю акторів. Такий підхід дозволяє виконувати одночасно сотні тисяч і навіть мільйони таких процесів, кожен з яких може мати скромні вимоги по пам'яті. Наприклад, при реалізації на Erlang мережевого чату структура програми може безпосередньо відображати одночасність дій користувачів з обміну повідомленнями шляхом запуску нових процесів. Ефективність передачі повідомлень зберігається і при збільшенні числа процесів, а вимоги до пам'яті мінімізуються за рахунок того, що полегшеними процесами управляє віртуальна машина, а не засоби операційної системи.

# Список використаних ресурсів

1. Erlang [з мережі] <http://ru.wikipedia.org/wiki/Erlang>
2. Центр суперкомп’ютерних досліджень [з мережі] <http://grid.kpi.ua/index.php/ru/national-resource-centre/10-centr-superkompyuternih-obchislen.html>
3. Erlang programing language [з мережі] <http://www.erlang.org/>
4. Ehabberd [з мережі] <http://ru.wikipedia.org/wiki/Ejabberd>
5. Ejabberd community site [з мережі] <http://www.ejabberd.im/>