Lab 02: Parallel Collection

Student ID	Name
66070286	ปนัสยา บุญประกอบ

Objectives:

1. (CLO4) สามารถอธิบาย และเขียนโปรแกรมเพื่อการประมวลผลขนานเชิงฟังก์ชัน (Functional Parallelism)

Ref.: https://docs.scala-lang.org/overviews/parallel-collections/overview.html

Prerequisite:

Create project (scala 2) > `sbt new scala/hello-world.g8`

แก้ไข build.sbt

1. **เลือก scala เวอร์ชั่นที่ต่ำกว่า 2.3**

```
scalaVersion := "2.12.18"
```

2. **import**

```
import scala.collection.parallel.immutable._
import scala.collection.parallel.mutable._
```

Exercise 1: Parallel Collection

1.1 Parallel Array

Code	Results
val pa = ParArray.tabulate(1000)(x => 2 * x + 1)	ParArray(1, 3, 5, 7, 9, 11, 13, 15, 17, …, 1999)
println(pa.map(x => (x - 1) / 2))	ParArray(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, …, 999)

เพิ่มเติมคำอธิบายตามความเข้าใจ

(1000) คือวนไป 1000 ครั้ง โดยเริ่มจาก 0-999
(x => 2 * x + 1) นำค่า x ชนิด Parallel Array ไปใส่ใน pa โดยที่ x จะเป็น 2 * x + 1 ทุกครั้งที่วนรอบ
pa.map(x => (x - 1) / 2) แมพสิ่งที่อยู่ใน pa ด้วยสูตร (x-1)/2

1.2 Parallel Vector

Code	Results
val pv = new ParVector[Int]	ParVector()
val pv = Vector(1,2,3,4,5,6,7,8,9).par	ParVector(1, 2, 3, 4, 5, 6, 7, 8, 9)
val pv2 = scala.collection.parallel.immutable.ParVector .tabulate(1000)(x => x) println(pv2.filter(_%2 == 0))	ParVector(0, 2, 4, 6, 8, 10, 12, 14, 16, …, 998)

เพิ่มเติมคำอธิบายตามความเข้าใจ

val pv = new ParVector[Int] คือการสร้างตัวแปร pv ชนิด Parallel Vector

val pv = Vector(1,2,3,4,5,6,7,8,9).par คือ การนำ Vector(1,2,3,4,5,6,7,8,9) แปลงเป็น parallel vector ด้วย .par จากนั้นเข้าไปใส่ใน pv ด้วยการประมวลผลแบบ parallel processing

val pv2 = scala.collection.parallel.immutable.ParVector.tabulate(1000)(x => x)

คือการเอาเลข 0-999 ใส่เข้าไปใน pv2 แบบ parallel vector

println(pv2.filter(_%2 == 0)) จะ print เลขที่หารด้วย 2 แล้วได้เศษ 0 (เลขคู่) ออกมา

1.3 Parallel Range

Code	Results
(1 to 3).par	ParRange(1, 2, 3)
(15 to 5 by -2).par	ParRange(15, 13, 11, 9, 7, 5)

เพิ่มเติมคำอธิบายตามความเข้าใจ

(1 to 3).par

(1 to 3) สร้าง Range(1, 2, 3) แล้วใช้ .par เพื่อทำให้เป็น Parallel Range

(15 to 5 by -2).par

สร้าง Range ที่เลขเริ่มตั้งแต่ 15-5 โดยจะ -2 ทุกครั้งที่วนรอบ แล้วใช้ .par เพื่อทำให้เป็น Parallel Range

Exercise 2: Non-determinism

non-determinism : ให้ผลลัพธ์ที่แตกต่างกันในแต่ละครั้งที่ป้อนข้อมูลแบบเดียวกัน

2.1 Side-Effecting Operations

Code	Results
var sum = 0	0
val list = (1 to 1000).toList.par	ParVector(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, … ,1000)
list.foreach(sum += _) println(sum)	465332
var sum = 0 list.foreach(sum += _) println(sum)	475174
var sum = 0 list.foreach(sum += _) println(sum)	449863

เพิ่มเติมคำอธิบายตามความเข้าใจ

sum = 0 เพื่อเก็บผลรวม

val list = (1 to 1000).toList.par .toList เพื่อสร้าง List และ .par เพื่อแปลงเป็น Parallel Vector

list.foreach(sum+=_) เพื่อบวกค่าใน list เข้ากับ sum

ที่ได้ผลลัพธ์ไม่เท่ากันในแต่ครั้ง เพราะทำงานแบบ parallel แล้วไม่มีการ synchronized กัน

2.2 Non-associative operations

Code	Results
val list2 = (1 to 1000).toList.par println(list2.reduce(_-_))	-134036
println(list2.reduce(_-_))	-99986
println(list2.reduce(_-_))	-35582

เพิ่มเติมคำอธิบายตามความเข้าใจ

สร้าง **val list2 = (1 to 1000).toList.par** เพื่อสร้าง list ที่มีเลข 1-1000 แบบ Parallel Vector

println(list2.reduce(_-_)) คือการ print ค่าของเลข 2 ตัว ใน list2 มาลบกัน ที่ได้ผลลัพธ์ไม่เท่ากันในแต่ละครั้งเพราะการลบคือ non-associative operation ใช้ parallel computation ไม่ได้

Code	Results
val strings = List("abc","def","ghi","jk","lmnop","qrs","tuv","wx","yz").par println(strings)	ParVector(abc, def, ghi, jk, lmnop, qrs, tuv, wx, yz)

val alphabet = strings.reduce(_+_) println(alphabet)	abcdefghijklmnopqrstuvwxyz

เพิ่มเติมคำอธิบายตามความเข้าใจ

val strings = List("abc","def","ghi","jk","lmnop","qrs","tuv","wx","yz").par

คือการสร้างตัวแปร List ที่ทำงานแบบ parallel
val alphabet = strings.reduce(_+_) คือการเอาค่า string มาต่อกัน
และทำกี่ครั้งก็จะได้คำตอบเหมือนเดิมเพราะเป็น associative operation ใช้ parallel computation ได้