**Weekly report**: MaRCS : Managing of Reputation in Crowdsourcing System

ผลงานช่วง วันจันทร์ที่ 14 มีนาคม 2559 ถึง วันเสาร์ที่ 19 กุมพาพันธ์ 2559

รายงาน วันเสาร์ที่ 19 กุมพาพันธ์ 2559

โดย

นางสาวนภวรรณ ดุษฎีเวทกุล เลขทะเบียนนักศึกษา 5509611637

นายวงศธร ทองถาวร เลขทะเบียนนักศึกษา 5509680061

1. **สิ่งที่ได้รับมอบหมายในช่วงที่แล้วและทำสำเร็จ**

* ทำ Presentation แสดงการทำงานของอัลกอริทึมต่างๆ
* สรุปผลการทดลองใหม่

1. **สิ่งที่ได้รับมอบหมายในช่วงที่แล้วและยังไม่ได้ทำ หรือทำแล้วติดปัญหา**

* Validate การทำงานใน NetLogo

1. **รายละเอียดเพิ่มเติม**

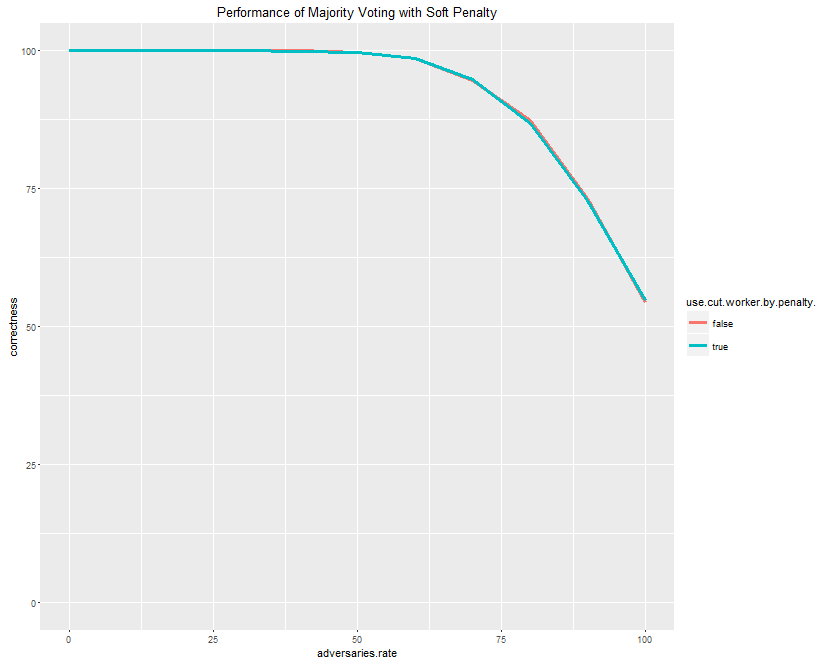
* สรุปผลการทดลอง (แก้ไข)

1. ผู้ใช้ปกติ

|  |
| --- |
| **ตัวแปรที่สนใจ**  อัตราส่วนของผู้ก่อกวน (0 – 100%)  **การตั้งค่า Parameter**  จำนวนผู้ใช้ทั้งหมด : 100 คน  โอกาสที่ผู้ใช้ปกติจะตอบค่าจริง : 70%  ประเภทของผู้ก่อกวน : ตอบตรงความกับความจริง  จำนวน Problem : 1 Problem  ลักษณะ Problem : น้ำไม่ท่วมเสมอ  Data Fusion Algorithm : Majority Voting  Reputation Algorithm : Soft Penalty  เปอร์เซ็นในการตัดผู้ใช้ : 10% |

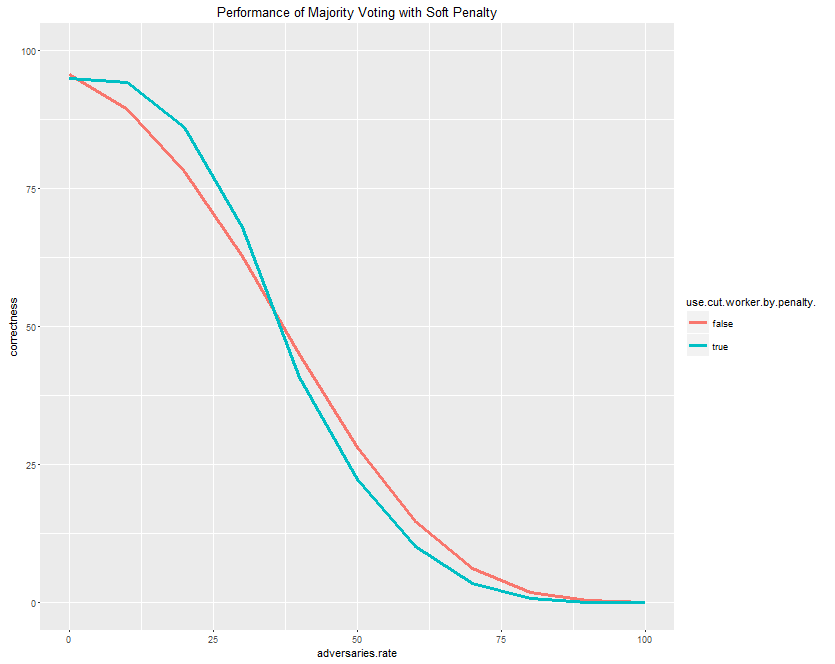
1. การจัดการความน่าเชื่อถือไม่สามารถเพิ่มความถูกต้องของระบบได้ เมื่อมีจำนวน Problem เป็น 1

|  |
| --- |
| **ตัวแปรที่สนใจ**  อัตราส่วนของผู้ก่อกวน (0 – 100%)  **การตั้งค่า Parameter**  จำนวนผู้ใช้ทั้งหมด : 100 คน  โอกาสที่ผู้ใช้ปกติจะตอบค่าจริง : 70%  ประเภทของผู้ก่อกวน : ตอบตรงความกับความจริง  จำนวน Problem : 1 Problem  ลักษณะ Problem : น้ำไม่ท่วมเสมอ  Data Fusion Algorithm : Majority Voting  Reputation Algorithm : Soft Penalty  เปอร์เซ็นในการตัดผู้ใช้ : 10% |



2. เมื่อจำนวน Problem มากขึ้น การจัดการความน่าเชื่อถือสามารถเพิ่มความถูกต้องของระบบได้มากขึ้น

|  |
| --- |
| **ตัวแปรที่สนใจ**  อัตราส่วนของผู้ก่อกวน (0 – 100%)  **การตั้งค่า Parameter**  จำนวนผู้ใช้ทั้งหมด : 100 คน  โอกาสที่ผู้ใช้ปกติจะตอบค่าจริง : 70%  ประเภทของผู้ก่อกวน : ตอบตรงความกับความจริง  จำนวน Problem : 100 Problem ( 10 x 10 )  ลักษณะ Problem : น้ำไม่ท่วมเสมอ  Data Fusion Algorithm : Majority Voting  Reputation Algorithm : Soft Penalty  เปอร์เซ็นในการตัดผู้ใช้ : 10% |



|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Adversaries Rate** | **0** | **10** | **20** | **30** | **40** | **50** | **60** | **70** | **80** | **90** | **100** |
| No Reputation | 95.727 | 89.311 | 78.093 | 62.730 | 44.994 | 28.025 | 14.619 | 6.146 | 1.880 | 0.345 | 0.000 |
| With Reputation | 95.037 | 94.283 | 86.064 | 67.863 | 40.723 | 22.291 | 10.123 | 3.463 | 0.691 | 0.010 | 0.000 |

จากผลที่ได้ จะเห็นได้ว่าเมื่อไม่มีผู้ก่อกวน วิธีที่ใช้ Reputation ความถูกต้องจะต่ำลงเล็กน้อย  
เนื่องจากวิธีที่มี Reputation จะตัดผู้ใช้ปกติบางส่วนออกไป แล้วทำให้ผู้ใช้ปกติเข้าถึง Problem ได้น้อยลง วิธี Reputation สามารถเพิ่มประสิทธิภาพของระบบ เมื่อมีอัตราส่วนของผู้ก่อกวนในระบบระดับหนึ่ง (10 – 30%) แต่เมื่ออัตราส่วนของผู้ก่อกวนมากกว่า 40% การ Reputation จะทำให้ระบบมีประสิทธิภาพลดลง

**ตัวอย่างสถานการณ์**

กำหนดให้ผู้ใช้ A,B,C เป็นผู้ใช้ปกติที่ตอบความจริงทั้งหมด และ D,E เป็นผู้ใช้ก่อกวนโดยตอบค่าตรงข้ามกับความจริง โดยทำการรายงานน้ำท่วมทั้งหมด 4 พื้นที่ ในการ Update ข้อมูลของ Datacenter แต่ละรอบจะไม่คิดคะแนนผู้ที่มี Penalty มากที่สุดในรอบนั้น และใช้ Reputation Algorithm คือ Soft Penalty

**การ Update ข้อมูลครั้งที่ 1**

|  |  |
| --- | --- |
| A,B,C,D | E |
|  |  |

ผู้ใช้ D จะถูก Penalty มากที่สุดเนื่องจากเป็นเสียงส่วนน้อย และในพื้นที่ขวาบนจะได้รับข้อมูลที่ผิพลาด เนื่องจากผู้ใช้ E เป็นคนเดียวที่อยู่ในพื้นที่นั้น

**การ Update ข้อมูลครั้งที่ 2**

|  |  |
| --- | --- |
| A,B,C,E |  |
| D |  |

เนื่องจาก D เป็นผู้ที่มีค่า Penalty มากที่สุด Datacenter จะไม่คิดคะแนนของ D

ดังนั้นเมื่อเราใช้ Reputation Algorithm เราจะสามารถจัดการกรณีที่ผู้ก่อกวนพยายามส่งข้อมูลในพื้นที่ที่มีผู้ใช้ปกติน้อยได้