อภินันทนาการ

รายงานการวิจัยฉบับสมบูรณ์

การวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithms

โดย

ผศ.ดร.จักรกฤษณ์ เสน่ห์ นมะหุต

ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ

คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร

สำนักหอสมุด มหาวิทยาลัยนเรศวร
วันลงทะเบียน 17 มิ.ย. 2558
เลขทะเบียน i-6783016
เลขเรียกหนังสือ ว QA
275
ง216ร
2557

กันยายน 2557

สัญญาเลขที่ R2557B079

รายงานการวิจัยฉบับสมบูรณ์

โครงการ การวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithms

ผู้วิจัย

ผศ.ดร.จักรกฤษณ์ เสน่ห์ นมะหุต

ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ

คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร

ได้รับทุนอุดหนุนการวิจัยจากงบประมาณแผ่นดิน

สนับสนุนโดยกองทุนวิจัยมหาวิทยาลัยนเรศวร

แบบสรุปย่อการวิจัย

เรื่อง

"การวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithms

Performance measurement and comparison of Name Matching Algorithms"

โดย

ผศ.ดร.จักรกฤษณ์ เสน่ห์ นมะหุต (หัวหน้าโครงการวิจัย)
ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร

ได้รับทุนอุดหนุนการวิจัยจากงบประมาณแผ่นดิน
สนับสนุนโดยกองทุนวิจัยมหาวิทยาลัยนเรศวร

กันยายน 2557

แบบสรุปย่อการวิจัย

1. รายละเอียดเกี่ยวกับโครงการวิจัย

1.1 ชื่อโครงการ

ชื่อโครงการวิจัย (ภาษาไทย) โปรแกรมค้นหาข้อมูลการท่องเที่ยวด้วยเสียง

(ภาษาอังกฤษ) **Tourist Information Searching System by Speech**

1.2.หัวหน้าโครงการ

ชื่อหัวหน้าโครงการ: ผศ.ดร.จักรกฤษณ์ เสน่ห์ นมะหุต

Asst. Prof. Dr. Chakkrit Snae Namahoot

หมายเลขบัตรประจำตัวประชาชน: 3210300198976

หน่วยงาน: ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยี
สารสนเทศ คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร

สถานที่ติดต่อ: ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยี
สารสนเทศ คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร
อ.เมือง จ.พิษณุโลก รหัสไปรษณีย์ 65000

หมายเลขโทรศัพท์: 0-5526-1000-4 ต่อ 3262-3

มือถือ 0833778080

โทรสาร: 0-55 69-3263

E-mail: chakkrits@nu.ac.th

2. ความสำคัญและที่มาของปัญหาการวิจัย

ชื่อและนามสกุล ได้มีการนำมาใช้ในหลายๆ กระบวนการในการบ่งชี้ตัวบุคคลหรือสืบค้นเกี่ยวกับครอบครัวของแต่ละบุคคล แต่ในทางกลับกัน ชื่อที่มีการสะกดผิดหรือเขียนผิดก็อาจก่อให้เกิดปัญหาในการบ่งชี้และค้นหาตัวบุคคลได้ เช่น การค้นหาทางเว็บไซต์หรือเหตุผลทางด้านความปลอดภัย **ซึ่งจะเห็นได้ว่าการบันทึกข้อมูลของชื่อนั้นหากเขียนแตกต่างกันเล็กน้อยหรือเกิดความผิดพลาดในการคัดลอกชื่อที่เคยได้ถูกบันทึกไว้ก่อนแล้ว อาจก่อให้เกิดความผิดพลาดในการระบุตัวบุคคลได้ เช่น ชื่อที่เขียนแตกต่างกันสองชื่ออาจบ่งชี้ถึงบุคคลคนเดียวกัน เป็นต้น**

จากแนวคิดข้างต้น จึงมีการที่จะเชื่อมโยงข้อมูลต่าง ๆ โดยใช้การจับคู่ชื่อให้มีความถูกต้องมากที่สุดเท่าที่จะเป็นไปได้ ถ้าหากมีชื่อที่ซ้ำกัน เช่น “John Smith” ก็จะมีการสร้างตัวบ่งชี้ตัวที่สอง

บทคัดย่อ

การวิจัยนี้ได้นำเสนอการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการที่แตกต่างกันทั้งหมด 3 วิธีการได้แก่การวัดประสิทธิภาพตามแนวคิดของ Laitและ Randell วิธีการ F-Measure และเทคนิค Clusteringโดยใช้เนมแมทชิ่งอัลกอริทึมจำนวน 12อัลกอริทึมซึ่งสามารถแบ่งได้ 2 กลุ่ม คือ กลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง ประกอบด้วย6 อัลกอริทึม ได้แก่ Soundex, Soundex2, Phonex, Metaphone, Simplex และ NYSIIS และกลุ่มที่มีการเปรียบเทียบกับค่าThreshold ประกอบด้วย6 อัลกอริทึม ได้แก่Guth, Levenshtein,ISG, LIG1, LIG2และ LIG3ซึ่งอัลกอริทึมในกลุ่มนี้ ผู้วิจัยได้ทำการหาค่า Threshold ที่เหมาะสมมาใช้ในการวัดประสิทธิภาพโดยการวัดประสิทธิภาพทั้ง 3 วิธีการได้นำรายชื่อของA Dictionary of English Surnames ของ Reaney and Wilsonทั้งหมด ซึ่งเป็นรายชื่อที่มีความถูกต้องและน่าเชื่อถือ (Ground Truth) มาใช้ในการวิจัยพร้อมทั้งการเปรียบเทียบเวลาในการประมวลผลและเวลาของวิธีการวัดประสิทธิภาพ ทั้ง 3 วิธีการ

การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randellและวิธีการ F-Measure นั้นสามารถวัดประสิทธิภาพได้ทั้ง 2 กลุ่ม(12 อัลกอริทึม)โดยจากผลการทดสอบพบว่า LIG2 มีประสิทธิภาพสูงที่สุดของทั้ง 2 วิธีการอย่างไรก็ตาม การวัดประสิทธิภาพด้วยเทคนิค Clustering จะสามารถวัดประสิทธิภาพได้เพียงกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียงเท่านั้น เนื่องจากกลุ่มที่การเปรียบเทียบกับค่า Threshold ไม่สามารถสร้างรายการชื่อที่จะนำมาจัดกลุ่ม (Cluster) ได้ โดยอัลกอริทึมที่มีประสิทธิภาพสูงของการทดสอบ คือ NYSIIS และ Simplex

ผลจากการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมพบว่าอัลกอริทึมในกลุ่มที่มีการเปรียบเทียบกับค่าThreshold ใช้เวลาในการประมวลผลน้อยกว่ากลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง เนื่องจากกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียงต้องเปลี่ยนชื่อเป็นรหัสเสียงทั้ง 2 ชื่อ แต่กลุ่มที่มีการเปรียบเทียบกับค่า Threshold เป็นการหาค่าเพียงครั้งเดียวและผลจากการหาเวลาของวิธีการวัดประสิทธิภาพพบว่า เทคนิค Clustering ใช้เวลาในการประมวลผลน้อยที่สุด

จากผลการศึกษานี้ จะเห็นได้ว่า เนมแมทชิ่งอัลกอริทึมแต่ละอัลกอริทึมมีจุดเด่นในแต่ละด้านไม่เหมือนกัน การประยุกต์ใช้เนมแมทชิ่งอัลกอริทึมควรมีการเลือกให้เหมาะสมกับงาน เช่น การค้นหาชื่อในงานทะเบียนต่าง ๆ อาจต้องการความถูกต้องในการค้นหา ซึ่งพิจารณาได้จากค่า Overall Accuracy และค่า F-Scoreโดยเนมแมทชิ่งอัลกอริทึมที่เหมาะสมคือ LIG2 และ NYSIIS

สำหรับงานที่ต้องการความเร็วในการสืบค้น ควรเลือกใช้อัลกอริทึมในกลุ่มที่มีการเปรียบเทียบกับค่า Thresholdเช่น Levenshtein, LIG3 และ LIG2

คำสำคัญ การวัดประสิทธิภาพ วิธีการจับคู่ชื่อวิธีเทคนิคการจัดกลุ่มและอัลกอริทึม

ABSTRACT

This research presents performance measurement and a comparison of 12 name matching algorithmsbased on three measurement methods: the concept of Lait and Randell, F-measure and clusteringtechniques.The algorithms can be classified into two groups: (1) group of coding names to soundof withSoundex, Soundex2, Phonex, Metaphone, Simplex and NYSIIS and (2) group of comparison with threshold valuesincludingGuth, Levenshtein, ISG, LIG1, LIG2 and LIG3 algorithms. Threshold values are used to optimizeperformance of algorithms. In measuring the performance of the three methodswe use a list of English Surnames (accurate and reliable) and compare the processing time to evaluate the performance.

Performance measurement is based on the concept of Lait and Randell, and F-Measure, which can measure the performance of the two groups. LIG2 is the most powerful of the two methods. Performance measurementby clustering techniques can be done only with group of coding name to sound and NYSIIS and Simplex are the most efficient. The processing time used by clustering techniques is less than that of the two other methods: the methods of Lait and Randell and F-measure

From the study, each name matching algorithm has different characteristics. Applying name matching algorithms to a specific task should consider suitable algorithms, such as searching names in the official registration may require more accuracy. Therefore, LIG2 and NYSIIS are the most suitable algorithm because they give highest accuracy by Lait and Randell and F-measure. Regarding processing time the name matching algorithms from thegroup of comparison with threshold value are the most suitable,especiallyLevenshtein, LIG3 and LIG2.

Keywords: Performance measurement, name matching methods,clustering techniques and algorithm

ขึ้นมา โดยอาจเป็นข้อมูลเกี่ยวกับการเกิด ซึ่งเป็นข้อมูลของแต่ละบุคคลและไม่สามารถเปลี่ยนแปลงได้ หรืออีกวิธีหนึ่งอาจนำระบบตัวเลขมาใช้ในการแก้ไขปัญหานี้ เช่น การใช้ตัวเลขบัตรประจำตัวประชาชน ซึ่งระบบตัวเลขทำได้ง่ายและสามารถทำการจัดการเป็นรายบุคคลได้ โดยมีตัวเลขของแต่ละบุคคลที่ไม่ซ้ำกัน อย่างไรก็ตามระบบตัวเลขบางครั้งก็มีข้อจำกัด เช่น โรงพยาบาลที่มีเลขประจำตัวของคนไข้ ซึ่งใช้ได้เฉพาะในโรงพยาบาลเท่านั้น ไม่สามารถใช้ได้อย่างกว้างขวางและไม่สามารถนำไปใช้ในโรงพยาบาลอื่นได้ นอกจากนี้ยังอาจเกิดความผิดพลาดขึ้นได้จากการบันทึกข้อมูล คัดลอกข้อมูล หรือกรอกข้อมูล ดังนั้นจึงต้องใช้วิธีการทางการบัญชีมาใช้เพื่อลดความผิดพลาดที่อาจเกิดขึ้นได้ เพื่อให้การบันทึกข้อมูลและเรียกข้อมูลสามารถทำได้อย่างสะดวกและมีคุณภาพ ซึ่งหนึ่งในวิธีการเหล่านั้นก็คือ การ Check-Digits (Hamming, 1986)เมื่อไม่สามารถใช้ตัวเลขหรือหมายเลขที่ไม่ซ้ำกันเพียงอย่างเดียวได้ การจับคู่ตัวแปรธรรมชาติ เช่น ชื่อ นามสกุล วันเกิด เพศ หรือข้อมูลเพิ่มเติมอื่นๆ เช่น ที่อยู่ พร้อมรหัสไปรษณีย์ สถานที่เกิดมาใช้ร่วมด้วย เมื่อเร็ว ๆ นี้ได้มีการแนะนำให้ใช้รหัสอย่างง่าย ๆ มาใช้ในการรักษาความปลอดภัยในสถานที่ต่าง ๆ เช่น สนามบินหรือด่านตรวจ ร่วมกับข้อมูลเฉพาะบุคคล (Biometric Maker Information) เช่น ลายนิ้วมือ (Snae and Brueckner, 2006)

"ชื่อ" ยังคงมีปัญหาในเรื่องการออกเสียงและการสะกดคำที่แตกต่างกัน ซึ่งหากรู้วิธีการที่ทำให้ทราบว่า ชื่อที่สะกดหรือออกเสียงต่างกันหมายถึงบุคคลเดียวกันได้หรือไม่ ก็จะสามารถแก้ไขปัญหานี้ได้"นามสกุล" เป็นตัวแปรที่พบบ่อยที่สุดแม้ว่าจะมีแนวโน้มที่จะมีการผันแปรในการสะกดคำมากกว่าคำอื่น ๆ ซึ่งมักจะมีข้อผิดพลาดเกิดขึ้น แต่สามารถระบุได้ง่าย รูปแบบชื่อยังเป็นปัญหาหลักในการระบุตัวบุคคลเพราะไม่ง่ายที่จะตรวจสอบว่ารูปแบบชื่อที่สะกดต่างกันเป็นของคน ๆ เดียวกันหรือคนละคน โดยเฉพาะในสังคมที่มีความหลากหลายทางวัฒนธรรม การมีแบบแผนการตั้งชื่อที่ต่างกัน ภาษาที่แตกต่างกัน และระบบการเขียนและความคิดของแต่ละบุคคลในการติดต่อกันทำให้เกิดความผิดพลาดในการระบุตัวบุคคลได้ โดยความคลาดเคลื่อนที่อาจเกิดขึ้นได้ สามารถแบ่งออกได้เป็น3 ประเภท(Snae and Brueckner, 2006)ดังนี้

1) ความคลาดเคลื่อนจากรูปแบบตัวอักษร

ปัญหานี้เกิดจากการใช้ตัวพิมพ์ใหญ่ เครื่องหมายวรรคตอนการเว้นวรรคคำขยายความและการย่อ

2) ความคลาดเคลื่อนจากรูปแบบการสะกดคำ

ความคลาดเคลื่อนนี้จะขึ้นอยู่กับการสันนิษฐานที่มาและชื่อเป้าหมายที่มีความแตกต่างกันเนื่องจากความผิดพลาดในการคัดลอก เช่น การออกเสียงที่ต่างออกไป รวมถึงการสะกดคำผิด และการพิมพ์ผิด สิ่งคลาดเคลื่อนเหล่านี้แม้จะไม่ได้ส่งผลถึงการออกเสียง แต่ก็ก่อให้เกิดปัญหาในการจับคู่ของชื่อได้

3) ความคลาดเคลื่อนจากรูปแบบการออกเสียง

ความคลาดเคลื่อนจากการออกเสียงเกิดจากการที่โครงสร้างของชื่อถูกเปลี่ยนแปลงไปด้วยการออกเสียง เช่น การสะกดคำว่า "Pooh" ในภาษาอังกฤษ แต่ในภาษาเยอรมันจะสะกดเป็น "Phu" แทน ซึ่งการออกเสียงที่เปลี่ยนแปลงไปนี้ อาจเกิดจากการได้ยินผิด จึงทำให้โครงสร้างของชื่อมีการเปลี่ยนแปลงไป หรือคำว่า "MAXIME" และ "MAXIMIEN" เป็นชื่อที่เกี่ยวข้องกันแต่มีการออกเสียงแตกต่างกันเป็นอย่างมาก ซึ่งจริงๆ แล้วรูปแบบการออกเสียงของชื่อนั้นสามารถแสดงได้หลายรูปแบบ เช่น อาจแสดงเป็น "ADELINE" หรืออีกแบบที่ได้ถูกย่อให้สั้นเป็น "LINE"

มีนักวิจัยจำนวนมากที่พัฒนา Name Matching Algorithm เพื่อแก้ปัญหาดังกล่าว แต่อย่างไรก็ตาม ในหนึ่งอัลกอริทึมไม่สามารถแก้ปัญหาดังกล่าวได้ทั้งหมด ซึ่ง Snaeและ Diaz (2002) ได้

พัฒนาอัลกอริทึม คือ LIGที่สามารถแก้ปัญหาดังกล่าวได้ ต่อมา Snae(2007) ได้ทำการเปรียบเทียบประสิทธิภาพของอัลกอริทึมของ Name Matching โดยใช้แนวคิดของ Laitและ Randell(1998) โดยใช้ฐานข้อมูลของ Reaneyและ Wilson (1997) ซึ่งมีการใช้ชื่อเพียงบางส่วนของฐานข้อมูลของ Reaneyและ Wilson เท่านั้น

ดังนั้นผู้วิจัยจึงต้องการทำการทดสอบName Matching Algorithm ทั้งหมด 12 อัลกอริทึม คือ Guth, Levenshtein, Soundex, Soundex2, Metaphone, Phonex, NYSIIS, Simplex, ISG, LIG1, LIG2, LIG3 และทำการวัดประสิทธิภาพและเปรียบเทียบName Matching Algorithm โดยใช้วิธีการหาค่า True Match, ค่า True Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998),วิธีการ F-measureตามแนวคิดของ Christen (2006), วิธีการจัดกลุ่ม และ Clusteringตามแนวคิดของ Snae(2006) และเวลาที่ใช้ในการประมวลผล Name Matching Algorithm โดยใช้ฐานข้อมูลชื่อทั้งหมดของ A Dictionary of English Surnames ของ Reaney and Wilson (1997)ซึ่งเป็นฐานข้อมูลที่มีความถูกต้อง(Ground truth)โดยส่วนใหญ่ในงานวิจัยอื่นที่เกี่ยวกับการวัดประสิทธิภาพและเปรียบเทียบ NameMatching Algorithm ได้ใช้ฐานข้อมูลที่มีการใช้งานจริงในองค์กรต่างๆซึ่งชื่อที่พ้องรูปหรือพ้องเสียงเกิดจากการสะกดผิดหรือเขียนผิดเกิดจากความผิดพลาดของคนกรอกข้อมูลในฐานข้อมูลในขณะนั้น แต่ฐานข้อมูลชื่อของ Reaney and Wilson จะเก็บเป็นชื่อของคนอังกฤษทั้งหมดที่ชื่อหลัก (Basename)สามารถมีการเขียนได้อย่าถูกต้องได้หลายแบบ (Variation) และในงานวิจัยนี้จะใช้ภาษา PHP ในการทดสอบ เนื่องจากภาษา PHP มีฟังค์ชันในการทำการกับสายอักขระ(String) ได้ดี และยังมีฟังค์ชันที่เกี่ยวกับ Name Matching Algorithm บางตัวอีกด้วย ส่วนในเรื่องของการวัดประสิทธิภาพและเปรียบเทียบในเรื่องของเวลาในการประมวลผลนั้น สามารถหาเวลาในการประมวลผลได้จากเครื่อง Server เนื่องจากภาษา PHP เป็นภาษาที่ทำงานแบบ Server-sideและยังสะท้อนแนวทางการทำงานในการค้นหาชื่อบุคคล ชื่อองค์กร (Snae and Brueckner, 2007) หรือสถานที่ท่องเที่ยว (Panawong and Snae, 2010) ที่มีการพิมพ์หรือสะกดผิด ในปัจจุบันที่ทำการค้นหาผ่าน Web Browser ต่าง ๆ

3. วัตถุประสงค์การวิจัย

1. เพื่อทดสอบName Matching Algorithms
2. เพื่อวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithms โดยมีหลักการวัดประสิทธิภาพดังนี้

2.1 วิธีการหาค่า True Match, ค่า True Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998)

2.2 วิธีการ F-measure ตามแนวคิดของ Christen (2006)

2.3 วิธีการจัดกลุ่ม และ Clustering ตามแนวคิดของ Snae (2006)

2.4 เวลาที่ใช้ในการประมวลผลของName Matching Algorithm

4. ระเบียบวิธีการวิจัย (โดยย่อ)

ในงานวิจัยนี้ได้นำName Matching Algorithm 12 อัลกอริทึมมาพัฒนาโปรแกรมการเปรียบเทียบชื่อสองชื่อว่ามีความคล้ายคลึงกันหรือไม่เพื่อใช้ในการวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithm โดยใช้วิธีการหาค่า True Match, ค่า True

Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998)วิธีการ F-measure ตามแนวคิดของ Christen (2006), วิธีการจัดกลุ่ม และ Clustering ตามแนวคิดของSnae (2006) และเวลาที่ใช้ในการประมวลผล Name Matching Algorithm โดยใช้ฐานข้อมูลชื่อทั้งหมดของA Dictionary of English Surnames ของ Reaney andWilson (1997)

5. ผลการดำเนินงานวิจัย

จากผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมที่สามารถวัด ประสิทธิภาพได้ทั้งหมด 12 อัลกอริทึม มีเพียง 2 วิธีการ ได้แก่ การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randellและวิธีการ F-Measure โดยผลการวัดประสิทธิภาพพบว่า LIG2 มีประสิทธิภาพสูงที่สุด ทั้ง 2 วิธีการ

การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมทั้ง 12 อัลกอริทึมโดยใช้การวัดประสิทธิภาพทั้งหมด 3 วิธีการ สรุปได้ว่าเนมแมทชิ่งอัลกอริทึมแต่ละอัลกอริทึมมีจุดเด่นในแต่ละเรื่องไม่เหมือนกัน เพราะฉะนั้น การเลือกใช้เนมแมทชิ่งอัลกอริทึมไปประยุกต์ใช้กับงานแต่ละงานควรเลือกให้เหมาะสม เช่น Search Engine ควรใช้อัลกอริทึมที่สามารถค้นหาชื่อออกมาให้ได้มากที่สุดซึ่งอาจพิจารณาจากค่า True Matches ดังนั้น เนมแมทชิ่งอัลกอริทึมที่เหมาะสม คือ Soundex, Soundex2, Phonexและ Levenshteinหรือการค้นหาชื่อในงานทะเบียนต่าง ๆ อาจต้องการความถูกต้องในการค้นหา ซึ่งพิจารณาได้จากค่า Overall Accuracy และค่า F-Scoreโดยเนมแมทชิ่งอัลกอริทึมที่เหมาะสมคือ LIG2และ NYSIIS

ในเรื่องของเวลาการประมวลผลพบว่า เนมแมทชิ่งอัลกอริทึมที่มีเวลาในการประมวลผลน้อยจะอยู่ในกลุ่มที่การเปรียบเทียบกับค่าThreshold ซึ่งมีเพียง Soundexและ Metaphoneเท่านั้นที่อยู่ในกลุ่มของการเปลี่ยนชื่อเป็นรหัสเสียง เนื่องจาก มีการใช้ฟังก์ชันของภาษา PHP จึงทำให้มีเวลาในการประมวลผลน้อยลง และการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมได้นำอัลกอริทึมที่มีเวลาในการประมวลผลน้อยที่สุดและมีประสิทธิภาพสูงมาใช้ในการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพซึ่งจากผลการทดลอง สรุปได้ว่า การวัดประสิทธิภาพที่มีเวลาน้อยที่สุด คือ การวัดประสิทธิภาพด้วยเทคนิค Clustering ดังนั้น ในการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมควรนำเทคนิค Clustering ไปใช้

6. ข้อเสนอแนะที่ได้จากการวิจัย

จากการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมทั้ง 12 อัลกอริทึมจะเห็นได้ว่าผลการทดสอบที่ได้มีความน่าเชื่อถือ และตรงตามขอบเขตที่ได้กำหนดไว้ แต่อย่างไรก็ตาม ข้อเสนอแนะที่ควรมีการปรับปรุงให้ดียิ่งขึ้น มีดังต่อไปนี้

1) ควรมีการวัดประสิทธิภาพในภาษาอื่น ๆ นอกเหนือจากภาษาอังกฤษเพิ่มมากขึ้นเช่น ภาษาไทย
2) ควรมีการนำภาษาคอมพิวเตอร์อื่นมาใช้ในการทดลองเพื่อผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมมีความน่าเชื่อถือมากยิ่งขึ้น เช่น ภาษา C

จากข้อเสนอแนะที่กล่าวมาข้างต้น สรุปได้ว่า การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทย อาจจะใช้รายชื่อที่มีความถูกต้องและน่าเชื่อถือ เช่น รายชื่อจากทะเบียนทะเบียนราษฎร์รายชื่อจาก Family Tree ของคนไทย เป็นต้นมาวัดประสิทธิภาพด้วยวิธีการที่มีความแตกต่างกันได้ ซึ่งการประยุกต์ใช้เนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทยนั้น Snaeและ Bruckner (2009) ได้พัฒนาเนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทย คือ Metasoundซึ่งนำไปใช้กับการตั้งชื่อและเปลี่ยนชื่อไทยให้สอดคล้องกับชื่อบิดาและมารดา หรือเปลี่ยนชื่อให้มีความคล้ายคลึงกับชื่อเดิม นอกจากนี้ Panawongและ Snae (2010) ได้พัฒนาระบบค้นหาสถานที่ท่องเที่ยวในประเทศไทยโดยได้มีการนำเนมแมทชิ่งอัลกอริทึมมาช่วยในการสืบค้นสถานที่ท่องเที่ยวที่มีการพิมพ์หรือสะกดที่คล้ายคลึงกัน เพื่อให้ได้ข้อมูลครอบคลุมมากยิ่งขึ้น

7. การนำไปใช้ประโยชน์

สามารถนำเอาอัลกอริทึมที่ดีหรือเหมาะสมที่สุดไปประยุกต์ใช้ในการค้นหาข้อมูลท่องเที่ยวที่มีการเขียนผิดหรือใกล้เคียงกันโดยผ่านเว็บแอพลิเคชันสำหรับการบริการข้อมูลแหล่งท่องเที่ยวและการบริการในพื้นที่จังหวัดที่อยู่ภายใต้การดูแลของการท่องเที่ยวแห่งประเทศไทย อีกทั้งยังสามารถนำไปประยุกต์ใช้ในโรงพยาบาลสำหรับตรวจสอบชื่อผู้ป่วยที่มีการเขียนผิดได้อีกด้วย และสามารถนำไปประยุกต์ใช้กับงานวิจัยอื่นๆ หรือ พัฒนาต่อยอดในการค้นหา family tree ประชากร ของประเทศไทยได้อย่างมีประสิทธิภาพ

บทที่ 1
บทนำ

ชื่อและนามสกุล ได้มีการนำมาใช้ในหลายๆ กระบวนการในการบ่งชี้ตัวบุคคลหรือสืบค้นเกี่ยวกับครอบครัวของแต่ละบุคคล แต่ในทางกลับกัน ชื่อที่มีการสะกดผิดหรือเขียนผิดก็อาจ ก่อให้เกิดปัญหาในการบ่งชี้และค้นหาตัวบุคคลได้ เช่น การค้นหาทางเว็บไซต์หรือเหตุผลทางด้านความปลอดภัย ซึ่งจะเห็นได้ว่าการบันทึกข้อมูลของชื่อนั้นหากเขียนแตกต่างกันเล็กน้อยหรือเกิดความผิดพลาดในการคัดลอกชื่อที่เคยได้ถูกบันทึกไว้ก่อนแล้ว อาจก่อให้เกิดความผิดพลาดในการระบุตัวบุคคลได้ เช่น ชื่อที่เขียนแตกต่างกันสองชื่ออาจบ่งชี้ถึงบุคคลคนเดียวกัน เป็นต้น

จากแนวคิดข้างต้น จึงมีการที่จะเชื่อมโยงข้อมูลต่าง ๆ โดยใช้การจับคู่ชื่อให้มีความถูกต้องมากที่สุดเท่าที่จะเป็นไปได้ ถ้าหากมีชื่อที่ซ้ำกัน เช่น "John Smith" ก็จะมีการสร้างตัวบ่งชี้ตัวที่สองขึ้นมา โดยอาจเป็นข้อมูลเกี่ยวกับการเกิด ซึ่งเป็นข้อมูลของแต่ละบุคคลและไม่สามารถเปลี่ยนแปลงได้ หรืออีกวิธีหนึ่งอาจนำระบบตัวเลขมาใช้ในการแก้ไขปัญหานี้ เช่น การใช้ตัวเลขบัตรประจำตัวประชาชน ซึ่งระบบตัวเลขทำได้ง่ายและสามารถทำการจัดการเป็นรายบุคคลได้ โดยมีตัวเลขของแต่ละบุคคลที่ไม่ซ้ำกัน อย่างไรก็ตามระบบตัวเลขบางครั้งก็มีข้อจำกัด เช่น โรงพยาบาลที่มีเลขประจำตัวของคนไข้ ซึ่งใช้ได้เฉพาะในโรงพยาบาลเท่านั้น ไม่สามารถใช้ได้อย่างกว้างขวางและไม่สามารถนำไปใช้ในโรงพยาบาลอื่นได้ นอกจากนี้ยังอาจเกิดความผิดพลาดขึ้นได้จากการบันทึกข้อมูล คัดลอกข้อมูล หรือกรอกข้อมูล ดังนั้นจึงต้องใช้วิธีการทางการบัญชีมาใช้เพื่อลดความผิดพลาดที่อาจเกิดขึ้นได้ เพื่อให้การบันทึกข้อมูลและเรียกข้อมูลสามารถทำได้อย่างสะดวกและมีคุณภาพ ซึ่งหนึ่งในวิธีการเหล่านั้นก็คือการ Check-Digits (Hamming, 1986)เมื่อไม่สามารถใช้ตัวเลขหรือหมายเลขที่ไม่ซ้ำกันเพียงอย่างเดียวได้ การจับคู่ตัวแปรธรรมชาติ เช่น ชื่อ นามสกุล วันเกิด เพศ หรือข้อมูลเพิ่มเติมอื่นๆ เช่น ที่อยู่พร้อมรหัสไปรษณีย์ สถานที่เกิดมาใช้ร่วมด้วย เมื่อเร็ว ๆ นี้ได้มีการแนะนำให้ใช้รหัสอย่างง่าย ๆ มาใช้ในการรักษาความปลอดภัยในสถานที่ต่าง ๆ เช่น สนามบินหรือด่านตรวจ ร่วมกับข้อมูลเฉพาะบุคคล (Biometric Maker Information) เช่น ลายนิ้วมือ (Snae and Brueckner, 2006)

"ชื่อ" ยังคงมีปัญหาในเรื่องการออกเสียงและการสะกดคำที่แตกต่างกัน ซึ่งหากรู้วิธีการที่**ทำให้ทราบว่า ชื่อที่สะกดหรือออกเสียงต่างกันหมายถึงบุคคลเดียวกันได้หรือไม่ ก็จะสามารถแก้ไขปัญหานี้ได้"นามสกุล" เป็นตัวแปรที่พบบ่อยที่สุดแม้ว่าจะมีแนวโน้มที่จะมีการผันแปรในการสะกดคำมากกว่าคำอื่น ๆ ซึ่งมักจะมีข้อผิดพลาดเกิดขึ้น แต่สามารถระบุได้ง่าย รูปแบบชื่อยังเป็นปัญหา**

2.2 วิธีการ F-measure ตามแนวคิดของ Christen (2006)

2.3 วิธีการจัดกลุ่ม และ Clustering ตามแนวคิดของ Snae (2006)

2.4 เวลาที่ใช้ในการประมวลผลของName Matching Algorithm

ขอบเขตของโครงการวิจัย

1. ทดสอบ Name Matching Algorithm ซึ่งประกอบไปด้วย Guth, LevenshteinSoundex, Soundex2, Phonex, Metaphone, Simplex, NYSIIS, ISG, LIG1, LIG2, และ LIG3โดยใช้ภาษา PHP
2. วัดประสิทธิภาพและเปรียบเทียบName Matching Algorithms โดยมีหลักการวัดประสิทธิภาพดังนี้
    2.1 วิธีการหาค่า True Match, ค่า True Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998)
    2.2 วิธีการ F-measure ตามแนวคิดของ Christen (2006)
    2.3 วิธีการจัดกลุ่ม และ Clustering ตามแนวคิดของSnae (2006)
    2.4 เวลาที่ใช้ในการประมวลผลของName Matching Algorithm
3. ฐานข้อมูลที่ใช้ในการเปรียบเทียบในการวิจัยนี้นำมาจาก A Dictionary of English Surnames (Reaney and Wilson, 1997)ทั้งหมด

บทที่ 2
เอกสารและงานวิจัยที่เกี่ยวข้อง

ในงานวิจัยนี้มีเอกสารที่เกี่ยวกับ Name Matching Algorithmดังนี้

- Name Matching Algorithm
- หลักการวัดประสิทธิภาพ
- A Dictionary of English Surnames
- ภาษา PHP

Name Matching Algorithm

Name Matching Algorithmเป็นอัลกอริทึมที่ใช้ในการเปรียบเทียบชื่อสองชื่อว่ามีความคล้ายกันหรือไม่(Match/No match) ซึ่งได้มีการแบ่งชนิดของName Matching Algorithmออกเป็น 4 ประเภท (Snae and Brueckner, 2006a) ได้แก่ กฎพื้นฐานของตัวสะกด (Spelling/string analysis based algorithms), กฎพื้นฐานของการออกเสียง (Phonetic/sound base algorithms), วิธีการรวมกัน(Composite methods) และวิธีการผสม (Hybrid Approaches)

กฎพื้นฐานของตัวสะกด (Spelling/stringanalysis based algorithms)

ใช้ในการเปรียบเทียบความคล้ายคลึงกันระหว่างตัวอักษรของชื่อ 2 ชื่อ ว่ามีหลักการเขียนเหมือนกันอย่างไร และผลของการเปรียบเทียบนี้จะออกมาในรูปว่าคล้ายกันหรือไม่ (Match/No match) ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Guthและ Levenshtein

A. Guth

วิธีการนี้เริ่มต้นจากการเปรียบเทียบตัวอักษรจากซ้ายไปขวา ไม่ว่าจะเป็นภาษาใดก็ตาม (Thaller, 1987)เป็นวิธีการเปรียบเทียบที่ไม่ซับซ้อนและให้ผลที่มีความน่าเชื่อถือ แต่มีข้อเสีย คือ เหมาะสำหรับใช้ในการเปรียบเทียบชื่อที่มีความยาวไม่มากนัก (De Brou and Olsen, 1986)

Guth Algorithm สามารถใช้ได้ในหลายภาษา (Guth, 1976) โดยวิธีการนี้เป็นวิธีการที่ง่าย และสามารถแก้ปัญหาการสะกดชื่อที่แตกต่างกันได้ ซึ่ง Guth Algorithm จะใช้วิธีการจับคู่ตัวอักษรในแต่ละตำแหน่งของชื่อทั้ง 2 ชื่อ โดยมีข้อดีทั้งหมด 4 ข้อ ดังนี้ (Guth, 1976)

- ไม่ได้ขึ้นอยู่กับการออกเสียง

- เป็นขั้นตอนที่มีความเป็นอิสระสามารถดัดแปลงได้ง่ายตามความต้องการ
- เมื่อใช้ในการเชื่ยมโยงข้อมูลสามารถรวมขั้นตอนได้ แต่ไม่ได้เปลี่ยนแปลงการบันทึกข้อมูล
- เป็นทางเลือกของการสะกดชื่อโดยระบุตำแหน่งของตัวอักษร

ขั้นตอนของ Guthสามารถให้ผลที่ไม่ถูกต้องระหว่างชื่อที่คล้ายคลึงกัน และจะเป็นปัญหาโดยเฉพาะอย่างยิ่งเมื่อเปรียบเทียบกับชื่อที่มีสระน้อย ๆ(De Brou and Olson,1986) ซึ่งขั้นตอนการตรวจสอบชื่อของ Guthเป็นการเปรียบเทียบทีละตัวอักษรตามตำแหน่งดังตารางที่ 1 โดยมีทั้งหมด 9 การทดสอบ โดยต้องมีตัวอักษรที่เหมือนกันในการทดสอบจาก 1 ใน 9 นี้ถึงทำให้ชื่อสองชื่อมีความคล้ายคลึงกันตามหลักการของ Guth Algorithm

ตารางที่ 1 การทดสอบตัวอักษรตามตำแหน่งของ Guth Algorithm

	Position in Name 1	Position in Name 2
Test 1	X	X
Test 2	X	X + 1
Test 3	X	X + 2
Test 4	X	X – 1
Test 5	X – 1	X
Test 6	X + 1	X
Test 7	X + 2	X
Test 8	X + 1	X + 1
Test 9	X + 2	X + 2

B. Levenshtein

เป็นการวัดค่าแตกต่างของจำนวนการแก้ไขระหว่างชื่อ 2 ชื่อว่ามีการแก้ไขกี่ครั้ง (Levenshtein, 1965) ซึ่งจำนวนครั้ง คือ Levenshtein Distance (LD)โดยจะใช้กระบวนการเพิ่ม การลด และการแทนที่ของตัวอักษรจากชื่อหนึ่งไปยังอีกชื่อหนึ่ง ตัวอย่างเช่น หากกำหนดเกณฑ์ความคล้ายกัน (Threshold) เท่ากับ 2 และเมื่อวัดความแตกต่างของชื่อทั้งสองแล้วพบว่ามีค่า LD น้อยกว่าหรือเท่ากับ 2 แสดงว่าว่าชื่อทั้งสองนี้ Match กัน แต่หากชื่อทั้งสองมีค่า LD มากกว่า 2 แสดงว่าชื่อทั้งสองไม่Match กันซึ่งตัวอย่างของค่า LD เป็นดังนี้

- ถ้าชื่อที่ 1(S1) คือ Robert และชื่อที่ 2(S2) คือ Robert ดังนั้น LD (S1 ,S2) = 0 เพราะไม่มีการเปลี่ยนแปลงของตัวอักษร

- ถ้าชื่อที่ 1(S1) คือ Robert และชื่อที่ 2(S2) คือ Robernดังนั้น LD (S1 ,S2) = 1 เพราะมีการเปลี่ยนแปลงตัวอักษรจาก S2 เป็น S1 คือจาก n เป็น t จำนวน 1 ครั้ง

Levenshtein Distance ถูกใช้อย่างแพร่หลายในชีววิทยาระดับโมเลกุล เช่น การตรวจการตรงกันของ DNA (Levenshtein, 1965)

วัตถุประสงค์ของการนำ LD มาใช้เพื่อให้รู้ว่าชื่อสองชื่อมีความใกล้เคียงหรือคล้ายกันหรือไม่ หากต้องการตรวจสอบว่าชื่อสองชื่อเหมือนกันสามารถกำหนดเกณฑ์ความคล้ายกัน (Threshold) ให้เป็นศูนย์ แต่ถ้ากำหนดเกณฑ์ความคล้ายมากกว่าศูนย์ เป็นการตรวจสอบชื่อที่มีความคล้ายกันโดยการ Levenshtein Distanceมีอยู่ 3 แบบด้วยกันดังนี้

- การแทนที่ (Insert)
- การเพิ่ม (Add)
- การลบ (Delete)

ซึ่งตัวอย่างของ Levenshtein Distanceแสดงดังต่อไปนี้

GLAVIN GLAWYN

- Match (G, G)
- Match (L, L)
- Match (A, A)
- Replace (W, V)
- Replace (Y, I)
- Match (N, N)

Threshold = 2

จากตัวอย่างนี้ทำให้ GLAVIN, GLAWYN ให้ผลการทดสอบว่ามีความคล้ายคลึงกัน

กฎพื้นฐานของการออกเสียง (Phonetic/sound base algorithms)

เป็นการใช้โครงสร้างของเสียงเป็นตัวเปรียบเทียบค่าความคล้ายคลึงกัน ซึ่งใช้ในการแก้ปัญหาชื่อที่มีเสียงเดียวกันแต่เขียนหรือสะกดต่างกัน โดยใช้วิธีการจัดกลุ่มตัวอักษรจัดให้เสียงเดียวกันอยู่ในกลุ่มตัวอักษรเดียวกัน ตัวอย่างเช่น ตัวอักษร Phใช้จัดอยู่ในกลุ่มตัวอักษร F หรือในภาษาไทย "จ ท ธ ต ฏ" ให้จัดอยู่ในกลุ่มแม่กด (ด) อาทิเช่น คำว่า กาจ กับ กาด จะออกเสียงเหมือนกัน ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Soundex, Soundex2, Metaphone, NYSIIS และ Phonex

C. Soundex

Soundex Algorithm ถูกออกแบบมาใช้สำหรับชื่อภาษาอังกฤษ (Winchester, 1970) Soundex Algorithm ได้ถูกนำไปใช้ เพื่อลดจำนวนระเบียนของการแต่งงานโดยการเข้ารหัสทั้งนามสกุลของสามีและนามสกุลเดิมของภรรยา และยังมีการนำอัลกอริทึมนี้ ไปใช้ในการเชื่อมโยงประชากรอีกด้วย แต่อย่างไรก็ตาม จะพบว่าการละเว้นการออกเสียงอาจจะทำให้ข้อมูลที่สมควร Match กลับได้ผลเป็นไม่ Match และข้อมูลที่ไม่ Match กลับให้ผลที่ Match กัน (Davey And Jarvis, 1990) ซึ่งข้อเสียที่พบจากการใช้ Soundex Algorithm คือ ไม่เหมาะกับชื่อที่สั้น หรือชื่อที่มีจำนวนสระมาก (Gill et, al., 1993)

Soundex Algorithm จะทำการเปลี่ยนชื่อเป็นรหัสของ Soundexซึ่งประกอบด้วย 1 ตัวอักษรและ 3 ตัวเลข โดยมีขั้นตอนการเปลี่ยนดังต่อไปนี้

1) เก็บตัวอักษรตัวแรกของ Input ชื่อไว้
2) ถ้าตัวอักษร A, E, I, O, U ไม่ใช้ตัวอักษรตัวแรกให้ตัดออกไป
3) จากข้อที่ 1 เปลี่ยนอักษรที่เหลือเป็นรหัส Soundexดังแสดงในตารางที่ 2

ตารางที่ 2 แสดงการเปลี่ยนอักษรเป็นรหัส Soundex

Letter	Number is coded to
B, P, F and V	1
C, G, J, K, Q, S, X and Z	2
D and T	3

Letter	Number is coded to
L	4
M and N	5
R	6

4) ผลลัพธ์ของรหัส Soundexต้องอยู่ในรูปของตัวอักษร1 ตัวและตามด้วยตัวเลข3 ตัว

5) จากข้อ 4 ถ้าผลลัพธ์ของรหัส Soundexน้อยกว่า 4 ตัวรหัส Soundexให้ใส่ศูนย์แทนจนครบรหัส แต่ถ้ารหัส Soundexมากกว่า 4 รหัสให้ตัดออก

ตัวอย่างการเปลี่ยนชื่อเป็นรหัส Soundex เช่น Goodlad มีรหัสเป็น G343 และ Goodlake เป็น G342

D. Soundex2

Soundex2 ได้ปรับปรุงมาจากอัลกอริทึม Soundex ซึ่งจะมีขั้นตอนเหมือนกับ Soundex Algorithm แต่จะมีการเปลี่ยนเพียงขั้นตอนที่ 3 โดยรหัสของ Soundex2 สามารถแสดงในตารางที่ 3

ตารางที่ 3 แสดงการเปลี่ยนอักษรเป็นรหัส Soundex2

Letter	Number is coded to
B and P	1
F and V	2
C, K and S	3
G and J	4
Q, X and Z	5
D and T	6
L	7

Letter	Number is coded to
M and N	8
R	9

ตัวอย่าง การเปลี่ยนชื่อเป็นรหัส Soundex2 เช่น Goodlad Goodlake และ Goodliff มีรหัสเป็น G234

E. Metaphone

ส่วนใหญ่จะใช้แก้ปัญหากับชื่ออังกฤษ โดยอัลกอริทึมมีขั้นตอนดังต่อไปนี้

1) ถ้ามีตัวเลขให้ตัดตัวเลขออก แล้วเปลี่ยนอักษรทุกตัวให้เป็นตัวใหญ่
2) ถ้า KN หรือ GN หรือ PM หรือ AE หรือ WR เป็นตัวแรกให้วางไว้ตำแหน่งแรก
3) ถ้า X เป็นตัวอักษรแรกให้เปลี่ยนเป็น S
4) ถ้า WH เป็นตัวอักษรแรกให้เปลี่ยนเป็น W
5) ถ้ามีอักษรที่เหมือนกันไม่ต้องเพิ่มรหัส
6) ถ้าตัวอักษร A, E, I, O, U และ Y ไม่ใช่ตัวอักษรตัวแรกให้ตัดออกไป
7) เปลี่ยนตัวอักษรให้เป็นรหัสของ Metaphoneดังแสดงในตารางที่ 4

ตารางที่ 4 แสดงการเปลี่ยนตัวอักษรเป็นรหัส Metaphone

ตัวอักษร	รหัส	เงื่อนไข
B	B	ยกเว้นถ้าอยู่ตำแหน่งสุดท้ายของคำหลังจากตัวอักษร "m"
C	X	(เสียง 'sh') ถ้า "-cia-" หรือ "-ch-"
	S	เป็น "-ci-", "-ce-" หรือ "-cy-"
		ถ้าไม่ออกเสียง "-sci-", "-sce-" หรือ "-scy-"
	K	มิฉะนั้น, รวมทั้ง "-sch-"

ตัวอักษร	รหัส	เงื่อนไข
D	J	ถ้าอยู่ใน "-dge-", "-dgy-", "-dgi"
	T	นอกนั้นเป็น
F	F	
G		ไม่ออกเสียงถ้ามันอยู่ใน "-gh-" และไม่อยู่ในตำแหน่งสุดท้าย หรือก่อนหน้าสระ
		ใน "-gn" หรือ "-gned"
		ใน "-dge-", อื่นๆ., ที่เหนือกว่ากฎ
	J	ถ้า "i", หรือ "e", หรือ "y" ก่อน, ถ้าไม่อยู่เป็นคู่ติดกัน "gg"
	K	นอกนั้นเป็น
H		ไม่ออกเสียงถ้าอยู่หลังสระและไม่มีสระตามมา
	H	นอกนั้นเป็น
J	J	
K		ถ้าไม่ออกเสียงใน "c"
	K	นอกนั้นเป็น
L	L	
M	M	
N	N	

ตัวอักษร	รหัส	เงื่อนไข
P	F	ถ้า "h" ก่อน
	P	นอกนั้นเป็น
Q	K	
R	R	
S	X	(sh) ถ้า "h" ก่อน หรือใน "-sio-" หรือ "-sia-"
	S	นอกนั้นเป็น
T	X	(sh) ถ้า "h" ก่อน หรือใน "-sio-" หรือ "-sia-"
	0 (Zero)	(th) ถ้า "h" ก่อน
		ถ้าไม่ออกเสียงใน "tch-"
	T	นอกนั้นเป็น
V	F	
W		ไม่ออกเสียงถ้าไม่ถูกตามโดยสระ
	W	ถ้าถูกตามโดยสระ
X	KS	
Y		ไม่ออกเสียงถ้าไม่ถูกตามโดยสระ
	Y	ถ้าถูกตามโดยสระ
Z	S	

ตัวอย่าง การเปลี่ยนชื่อเป็นรหัส Metaphone เช่น Goodlad, Goodlake และ Goodliff มีรหัสเป็น KTLT, KTLK และ KTLF ตามลำดับ

F. Phonex

เกิดจากการรวมกันของสองกระบวนการ คือ Soundexและ Metaphone ซึ่งกระบวนการนี้ พบว่า สามารถทำงานได้ดีเมื่อนำไปใช้กับชื่อในภาษาอังกฤษ (Lait and Randell, 1998) ขั้นตอนของ Phonexในการแปลงชื่อให้เป็นรหัส 4 ตัวอักษรมีดังต่อไปนี้

- ลบตัวอักษร S ที่ต่อท้ายออก
- ถ้ามี KN หรือ WR ขึ้นต้นให้ตัดตัวอักษรแรกทิ้งไป
- ถ้ามีตัวอักษรดังต่อไปนี้ขึ้นต้นให้เปลี่ยนดังนี้
    - ตัวอักษร H ให้ตัดทิ้ง
    - ตัวอักษร E I O U Y ให้เปลี่ยนเป็น A
    - ตัวอักษร K และ Q ให้เปลี่ยนเป็น C
    - ตัวอักษร P ให้เปลี่ยนเป็น B
    - ตัวอักษร J ให้เปลี่ยนเป็น G
    - ตัว PH และ V ให้เปลี่ยนเป็น F
    - ตัว Z ให้เปลี่ยนเป็น S

ให้เก็บตัวอักษรตัวแรกไว้แล้วลบตัวอักษร A, E, H, I, O, U, W, Y ในตำแหน่งอื่นออก กำหนดหมายเลขดังต่อไปนี้ให้ตัวอักษรที่เหลือหลักตัวแรก

- B, F, P, V เปลี่ยนเป็น 1
- C, G, J, K, Q, S, X, Z เปลี่ยนเป็น2
- D, T เปลี่ยนเป็น 3 ถ้าไม่ตามด้วย C
- L เปลี่ยนเป็น 4 ถ้าไม่ตามด้วยสระและเป็นตัวอักษรสุดท้าย
- **M, N เปลี่ยนเป็น 5 ข้ามไปพิจารณาตัวอักษรถัดไปถ้าตัวอักษรถัดไปเป็น D หรือ G**
- **R เปลี่ยนเป็น 6 ถ้าไม่ตามด้วยสระและเป็นตัวอักษรสุดท้าย**
- **ข้ามตัวอักษรปัจจุบันถ้ามีหมายเลขเดียวกับตัวอักษรก่อนหน้า**

- รหัสของ Phonexต้องมีลักษณะ คือ ตัวอักษรตามด้วยตัวเลข 3 ตัว
- ถ้ารหัสมีตัวเลขไม่ถึง 3 ตัวให้เติมด้วย 0 จนครบ และถ้ามีตัวเลขครบ 3 ตัวแล้วยังมีตัวอักษรที่ยังไม่ได้เปลี่ยนให้เปลี่ยนที่ตัวเลขสุดท้าย

ตัวอย่างการเปลี่ยนชื่อเป็นรหัส Phonex เช่น Goodlad, Goodlake และ Goodliff มีรหัสเป็น G343, G342 และ G341ตามลำดับ

G. NYSIIS

เป็นวิธีการทางตัวอักษรที่ง่ายและให้ผลรหัสอัลกอริทึมเหมือนกับ Soundex อย่างไรก็ตาม NYSIIS ต่างจาก Soundexตรงที่มันเก็บข้อมูลเกี่ยวกับตำแหน่งของสระในการเข้ารหัสของคำโดยแปลงสระทั้งหมดให้เป็นตัวอักษร A (Gill et al., 1993) โดยวิธีการเข้ารหัสเบื้องต้นของ NYSIIS มีดังนี้

- ใช้ตัวอักษรตัวแรกของชื่อมาเป็นรหัสตัวแรกของ NYSIIS
- ลบสระที่ซ้ำกันให้เหลือเพียงหนึ่งตัว
- ถ้าชื่อประกอบด้วยสระครบทุกตัว (AEIOU) ให้ตัดสระทิ้ง
- แทนค่าสระด้วยตัว A
- ถ้าตัวอักษรตัวสุดท้ายเป็นสระ ให้ตัดทิ้ง
- ตัวอักษรที่เหลือให้เปลี่ยนเป็นรหัส NYSIIS

ตัวอย่าง การเปลี่ยนชื่อเป็นรหัส NYSIIS เช่น Goodlad, Goodlake และ Goodliff มีรหัสเป็น GADLAD, GADLAC และ GADLAF ตามลำดับ

วิธีการรวมกัน (Composite methods)

เป็นวิธีการรวมกันระหว่างกฎพื้นฐานของตัวสะกด (Spelling/string analysis based algorithms) และกฎพื้นฐานของการออกเสียง (Phonetic/sound base algorithms) ซึ่งการรวม 2 วิธีนี้เป็นการแก้ปัญหาของชื่อเกี่ยวกับคำพ้องรูปพ้องเสียงหรือคำที่เขียนเหมือนกันแต่ออกเสียงต่างกันและคำที่เขียนต่างกันแต่ออกเสียงเหมือนกัน ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Simplex และ ISG

H. Simplex

เป็นวิธีที่มีการปรับปรุงจากวิธี NYSIIS และ Soundex Algorithm ซึ่งจะเพิ่มจำนวนการ Match มากขึ้นจากเดิม โดยการแก้ไขส่วนใหญ่เน้นไปที่การแก้ไขตัวอักษร การเข้ารหัสของ Simplex มีพื้นฐานมาจาก NYSIIS โดยการเปลี่ยนสระ "AIEOUY"เป็น A ทั้งหมด

- ทำตามขั้นตอนข้อ 1-6 ของ NYSIIS Algorithm
- เข้ารหัสตัวอักษรตัวสุดท้ายและตัวอักษรตัวอื่นๆ ดังต่อไปนี้
- สำหรับตัวอักษรตัวสุดท้าย เข้ารหัสดังนี้
  - เปลี่ยน X เป็น C
  - เปลี่ยน V เป็น F
  - เปลี่ยน SE, ES, AS, CE เป็น S
  - เปลี่ยน YE, EE, IE, EY, EI เป็น Y
  - เปลี่ยน NT, ND เป็น N
- สำหรับตัวอักษรตัวอื่นๆ เข้ารหัสดังนี้
  - เปลี่ยน PH เป็น F
  - เปลี่ยน JH เป็น J
  - เปลี่ยน HN เป็น N
  - เปลี่ยน TH เป็น T
  - เปลี่ยน RH, RM เป็น R
  - เปลี่ยน WR เป็น R
  - เปลี่ยน WM, WN, M เป็น N
- ถ้ามี K ตามด้วย N ให้เปลี่ยนเป็น N แต่ถ้า K ตามด้วยตัวอักษรอื่นที่ไม่ใช่ N ให้เปลี่ยนเป็นตัว C
- ถ้ามี Z เป็นตัวอักษรตัวสุดท้ายให้เปลี่ยน Z เป็น S แต่ถ้าไม่ใช่ Z เป็นตัวสุดท้ายให้คงตัวอักษรเดิมไว้
- ถ้ามี Y เป็นทั้งตัวอักษรตัวแรกและตัวสุดท้าย ให้คงเดิม แต่ถ้าเป็นตัวอักษรอื่นให้เปลี่ยนเป็น A

ตัวอย่าง การเปลี่ยนชื่อเป็นรหัส Simplex เช่น Goodlad, Goodlake และ Goodliff มีรหัสเป็น GADLAD, GADLACA และ GADLAF ตามลำดับ

I. ISG

เป็นวิธีที่อธิบายโดย Bouchard and Poyez (1980) ซึ่งจะใช้ในการวัดหรือเปรียบเทียบค่าความคล้ายคลึงกันระหว่างชื่อโดยมีความละเอียดตั้งแต่ 0-1 แต่ข้อเสียของวิธีนี้คือ ไม่สามารถใช้ได้กับชื่อเล่นที่ถูกตัดมาจากชื่อเต็มเพราะจะทำให้มีค่า ISG ลดลง เช่น ชื่อเล่น Alex ซึ่งมากจากชื่อจริงว่า Alexander การวัดค่าความคล้ายคลึงนั้นจะใช้สูตรในการคำนวณดังต่อไปนี้ (Bouchard and Poyez, 1980)

$$ISG = \frac{I}{I + D}$$

เมื่อ I คือ จำนวนตัวอักษรที่เหมือนกันระหว่างชื่อ

D คือ จำนวนตัวอักษรที่แตกต่างกันระหว่างชื่อ

วิธีการผสม(Hybrid approaches)

เป็นการใช้ค่าความน่าจะเป็นในการวัดค่าความคล้ายคลึงของชื่อ (Snae and Diaz, 2002) ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ LIG1, LIG2, LIG3 (ย่อมากจาก Levenshtein Index of Similarity and Guth)

J. LIG1

เป็นการใช้หลักการรวมกัน (Composite) โดยรวมเอาอัลกอริทึมทั้ง 3 วิธีเข้าด้วยกัน คือ Levenshtein,ISG และ Guth (Snae and Diaz, 2002) มาใช้ในการค้นหาชื่อที่คล้ายคลึงกัน (Snae at al., 2006) ซึ่งมีข้อดี คือ ทำให้ค่าความถูกต้องในการค้นหาชื่อที่คล้ายหรือชื่อที่เหมือนกันมีมากขึ้น สามารถใช้ได้ดีกับชื่อในหลายๆ ประเทศ และใช้หลักการความน่าจะเป็นในการวัดค่าความคล้ายคลึงของชื่อซึ่งมีขั้นตอนดังต่อไปนี้

- เริ่มต้นจากการหาจำนวนตัวอักษรและตำแหน่งของชื่อ
- คำนวณหาจำนวนตัวอักษรที่แตกต่างกันของชื่อทั้งสอง โดยตัวอักษรต้องตรงตามตำแหน่ง
- ถ้ามีตัวอักษรเหมือนกันให้เพิ่มค่าความเหมือน
- ถ้าไม่เหมือนกันให้นำตัวอักษรของชื่อแรกไปเปรียบเทียบกับตัวอักษรทุกๆ ตัวในชื่อที่ 2 ถ้าพบว่าเหมือนกัน ให้เพิ่มค่าความเหมือน แต่ถ้าไม่พบตัวอักษรที่เหมือนกันให้เพิ่มค่าความต่าง
- คำนวณหาค่า LIG1 จากสูตรคำนวณดังต่อไปนี้ (Snae and Diaz, 2002)

$$LIG1 = \frac{I(G)}{I(G) + D}$$

เมื่อ $I(G)$ คือ จำนวนตัวอักษรที่เหมือนกันระหว่างชื่อ

D คือ จำนวนตัวอักษรที่แตกต่างกันระหว่างชื่อ

K. LIG2

เป็นการใช้สูตรของ LIG2 เพื่อใช้ในการคำนวณหาค่าความคล้ายคลึงของชื่อ ซึ่งมีขั้นตอนดังต่อไปนี้

- เริ่มต้นจากการหาจำนวนตัวอักษรและตำแหน่งของชื่อ
- คำนวณหาจำนวนตัวอักษรที่แตกต่างกันของชื่อทั้งสอง โดยตัวอักษรต้องตรงตามตำแหน่ง
- ถ้ามีตัวอักษรเหมือนกันให้เพิ่มค่าความเหมือน แต่ถ้าไม่เหมือนกันให้นำตัวอักษรของชื่อแรกไปเปรียบเทียบกับตัวอักษรทุกๆ ตัวในชื่อที่ 2 ถ้าพบว่าเหมือนกัน ให้เพิ่มค่าความเหมือน แต่ถ้าไม่พบตัวอักษรที่เหมือนกันให้เพิ่มค่าความต่าง
- หาค่า C (Cost of levenshtein distance)
- คำนวณหาค่า LIG2 จากสูตร

$$LIG2 = \frac{I}{I + C}$$

เมื่อ I คือ จำนวนตัวอักษรที่เหมือนกันระหว่างชื่อ

C คือ ฟังก์ชั่นการคำนวณหาค่าระยะทางที่แตกต่างระหว่างชื่อ 2 ชื่อโดยใช้หลักการของ Levenshtein (1965) และฟังก์ชั่นการคำนวณความคล้ายคลึงของชื่อ

- ถ้าค่า LIG2 > 0.5 แสดงว่าชื่อทั้งสอง Match กัน

L. LIG3

LIG3 มีขั้นตอนการคำนวณในลักษณะเดียวกับ LIG2 แต่เพียงเปลี่ยนสูตรการคำนวณในขั้นตอนที่ 5 ของ LIG2 เป็นดังนี้

$$\text{LIG3} = \frac{2I}{2I + C}$$

เมื่อ I คือ จำนวนตัวอักษรที่เหมือนกันระหว่างชื่อ

C คือ ฟังก์ชั่นการคำนวณหาค่าระยะทางที่แตกต่างระหว่างชื่อ 2 ชื่อโดยใช้หลักการของ Levenshtein (1965) และฟังก์ชั่นการคำนวณความคล้ายคลึงของชื่อ

วิธีการวัดประสิทธิภาพ

วิธีการหาค่าTrue Match, ค่าTrue Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998)

วิธีการวัดประสิทธิภาพโดยใช้หลักการของ Laitและ Randell (1998) คือการหาจำนวนของ True matches ซึ่งเป็นจำนวนของชื่อที่ Match ได้ถูกต้องโดยใช้สูตรการคำนวณดังต่อไปนี้

$$\text{True matches} = \frac{a}{v} * 100$$

เมื่อ a คือ จำนวนชื่อที่ Match ได้ถูกต้อง

v คือ จำนวนชื่อที่มีความหมายเดียวกัน

และจำนวนของ True mismatches ซึ่งเป็นค่าที่แสดงถึงจำนวนของชื่อที่ Match แต่ในความเป็นจริงต้องไม่ Match หาได้จาก

$$\text{True mismatches} = \frac{k - a}{k} * 100$$

เมื่อ a คือ จำนวนชื่อที่ Match ได้ถูกต้อง

k คือ จำนวนชื่อที่ Match ทั้งหมด

ส่วนค่าความถูกต้อง (Overall accuracy)ที่แสดงถึงประสิทธิภาพของ Name Matching Algorithm แต่ละตัวมีสูตรดังต่อไปนี้

$$\text{Overall accuracy} = \frac{a}{k} * 100$$

เมื่อ a คือ จำนวนชื่อที่ Match ได้ถูกต้อง

k คือ จำนวนชื่อที่ Match ทั้งหมด

วิธีการ F-measure ตามแนวคิดของChristen (2006)

วิธีการวัดประสิทธิภาพโดยใช้หลักการของ Christen (2006) หรือที่เรียกว่า f-score โดยใช้สูตรการคำนวณดังต่อไปนี้

$$\text{Precision} = \frac{TP}{TP + FP}$$

เมื่อ TP คือ จำนวนชื่อที่ควรจะ Match และ Match (True Positive)

FP คือ จำนวนชื่อที่ควรจะไม่ Match แต่ Match (False Positive)

$$\text{Recall} = \frac{TP}{TP + FN}$$

เมื่อ TP คือ จำนวนชื่อที่ควรจะ Match และ Match (True Positive)

FN คือ จำนวนชื่อที่ควรจะ Match แต่ไม่ Match (False Negative)

$$f - score = 2 * (\frac{P * R}{P + R})$$

เมื่อ P คือ ค่า Precision

R คือ ค่า Recall

วิธีการจัดกลุ่มและClustering ตามแนวคิดของSnae (2006)

เป็นเทคนิคสำหรับการวิเคราะห์ข้อมูลทางสถิติ ซึ่งถูกใช้ในสาขาต่างๆ รวมทั้งการเรียนรู้ของเครื่องจักรกล (Machine learning), เหมืองข้อมูล (Data Mining) และเป็นการจำแนกหนวดหมู่ของสิ่งที่คล้ายกันให้อยู่ด้วยกัน หรือเป็นการแบ่งหมวดหมู่ของข้อมูลให้อยู่เป็นกลุ่มย่อยๆ ซึ่งเรียกว่า Cluster **ดังนั้นข้อมูลในแต่ละกลุ่มย่อยๆ จะมีคุณสมบัติที่คล้ายคลึงกันหรือเหมือนกัน เทคนิค** Clustering นั้นถูกใช้กันอย่างแพร่หลายในหลายสาขาวิชา เช่น Statistics, Machine learning และ Data Mining

วัตถุประสงค์ของการจัดกลุ่ม คือการนำเสนอข้อมูลที่ไม่เหมือนกันไปสู่ในรูปแบบที่สามารถจัดการได้ง่ายโดยใช้หลักการแบ่งแยกให้เป็นหมวดหมู่ (Van Ooyen, 2001) ตัวอย่างเช่น Reaney and Wilson (1997) ได้จัดหมวดหมู่ของชื่อสกุลภาษาอังกฤษ และชื่อที่มีความเพี้ยนคล้ายคลึงกันในรูปแบบพจนานุกรม

Snae (2006)กล่าวว่าเทคนิคClusteringเป็นการนำรายการชื่อสองรายการมาเปรียบเทียบความคล้ายคลึงกัน มีขั้นตอนการดำเนินการ 2 ขั้นตอน คือ ขั้นตอนการจัดกลุ่มซึ่งเป็นวิธีการแบ่งกลุ่มของชื่อโดยให้ชื่อเดียวกันอยู่ในกลุ่มเดียวกันและขั้นตอนการสร้าง Cluster ซึ่งเป็นการวัดความคล้ายคลึงกันของรายการชื่อ

ตารางที่ 5 แสดงตัวอย่างการสร้าง Cluster

Name	R&W	Soundex	Soundex(Sort)	Group2	Cluster
Goodlad	1	G343	C342	1	1
Goodlatte	1	G343	G340	2	
Goodlet	1	G343	G341	3	
Goodlett	1	G343	G341	3	
Goodlake	2	G342	G342	4	2
Goodluck	2	G342	G342	4	
Gullick	2	G420	G343	5	
Gulick	2	G420	G343	5	
Cutlack	2	C342	G343	5	
Goodlamb	3	G345	G343	5	3

Name	R&W	Soundex	Soundex(Sort)	Group2	Cluster
Goodland	4	G345	G343	5	
Goodlet	5	G343	G343	5	
Goodley	6	G340	G345	6	4
Goodliff	7	G341	G345	6	
Goodliffe	7	G341	G350	7	5
Goodlud	8	G343	G352	8	6
Goodman	9	G350	G356	9	7
Goodner	10	G356	G363	10	
Goodness	11	G352	G363	10	8
Goodram	12	G365	G363	10	
Goodread	13	G363	G363	10	
Goodred	13	G363	G365	11	
Goodreds	13	G363	G420	12	9
Goodered	13	G363	G420	12	

เมื่อสร้างCluster เสร็จแล้วนำมาคำนวณโดยใช้สูตรดังต่อไปนี้ซึ่งให้ค่าตั้งแต่ 50% จึงถึง 100%

$$ch = \left(\frac{n+k}{2n}\right) * 100$$

เมื่อ ch คือ ค่าของ Cluster

n คือ จำนวนชื่อทั้งหมดในกลุ่ม

k คือ จำนวนชื่อลำดับที่ 2 ที่ใหญ่ที่สุดของ Cluster

การคำนวณหาประสิทธิภาพของ Name Matching Algorithm โดยวิธีการจัดกลุ่ม และ Clustering มีอยู่2 วิธี (Snae, 2006) คือ Included single clusterและ Ignored single clusters

ซึ่งในการคำนวณหาประสิทธิภาพของ Name Matching Algorithm แบบ Included single clusterระหว่างสองรายการชื่อหาได้จากสูตรคำนวณดังต่อไปนี้

$$i = \sum_{i=1}^{I} \{\frac{ch(i)}{I}\}$$

เมื่อ i คือ ประสิทธิภาพของ Name Matching Algorithm

ch คือ ค่าของ Cluster

I คือ จำนวน Cluster ทั้งหมด

สำหรับ Included single cluster นั้นเมื่อมี Cluster ใดที่มีเพียงชื่อเดียวจะทำให้ประสิทธิภาพของ Name Matching Algorithm เพิ่มมากขึ้น เนื่องจากค่า chของ Cluster นั้นจะเป็น 100 % ซึ่งในตัวอย่างคือ Cluster 5, 6 และ 7ซึ่งถ้ามี Cluster ที่มีชื่อเดียวจำนวนมากก็จะทำให้ประสิทธิภาพของ Name Matching Algorithm มีความผิดพลาดได้ดังนั้นการคำนวณหาประสิทธิภาพของ Name Matching Algorithm แบบIgnored single cluster สามารถที่จะสามารถช่วยให้ค่าความคล้ายคลึงมีความถูกต้องมากขึ้น โดยใช้สูตรในการคำนวณดังต่อไปนี้

$$i = \sum_{i=1}^{I} \sum_{j=1}^{J} \{\frac{ch(i) - ig(j)}{I - J}\}$$

เมื่อ ig คือ m*100 (m คือ จำนวน single cluster)

I คือ จำนวน Cluster ทั้งหมด

J คือ จำนวน Single Cluster

ดังนั้นประสิทธิภาพของ Name Matching Algorithm แบบIncluded single cluster และแบบ Ignored single cluster ตามตัวอย่างดังแสดงในตารางที่ 6

ตารางที่ 6 แสดงตัวอย่างผลของ Included single cluster และ Ignored single cluster

All cluster(I=9)	n	k	2n	ch(i)(%)	ig(i)(%)
Cluster 1	4	2	8	75	0
Cluster 2	5	3	10	80	0
Cluster 3	3	1	6	66.67	0
Cluster 4	2	1	4	75	0
Cluster 5	1	1	2	100	100
Cluster 6	1	1	2	100	100
Cluster 7	1	1	2	100	100
Cluster 8	4	1	8	62.5	0
Cluster 9	3	2	6	83.33	0
				742.50	300.00

ประสิทธิภาพของ Name Matching Algorithm แบบ Included single cluster คือ

742.50/9= 82.5%

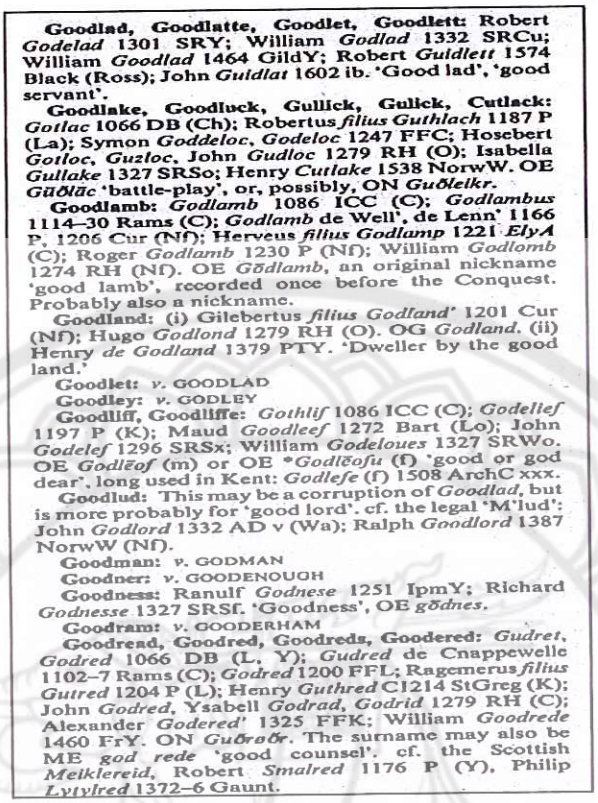
ประสิทธิภาพของ Name Matching Algorithm แบบ Ignored single cluster คือ

(742.50-300)/(9-3) = 73.75%

A Dictionary of English Surnames

พจนานุกรมนามสกุลภาษาอังกฤษ(Raney and Wilson, 1997)เป็นพจนานุกรมที่ระบุและให้ความรู้เกี่ยวกับต้นกำเนิดของสกุลภาษาอังกฤษซึ่งเป็นประวัติที่ยาวนานศตวรรษการสำรวจครอบคลุมถึงเรื่องที่น่าสนใจ มีข้อมูลเกี่ยวกับนามสกุลประมาณ16,000 นามสกุล ซึ่งสามารถอ้างอิงทางประวัติศาสตร์และภาษาศาสตร์ที่มีคุณค่าเกี่ยวกับที่มาของนามสกุล

รายการในพจนานุกรมแต่ละรายการจะประกอบไปด้วย Base name แล้วตามด้วย Variation name พร้อมทั้งคำอธิบายรายละเอียดในแต่ละรายการ ดังแสดงในรูปที่ 1



Goodlad, Goodlatte, Goodlet, Goodlett: Robert *Godelad* 1301 SRY; William *Godlad* 1332 SRCu; William *Goodlad* 1464 GildY; Robert *Guidlett* 1574 Black (Ross); John *Guidlat* 1602 ib. 'Good lad', 'good servant'.

Goodlake, Goodluck, Gullick, Gulick, Gutlack: *Gotlac* 1066 DB (Ch); Robertus *filius Guthlach* 1187 P (La); Symon *Goddeloc*, *Godeloc* 1247 FFC; Hosebert *Gotloc*, *Guzloc*, John *Gudloc* 1279 RH (O); Isabella *Gullake* 1327 SRSo; Henry *Cutlake* 1538 NorwW. OE *Gūðlāc* 'battle-play', or, possibly, ON *Guðleikr*.

Goodlamb: *Godlamb* 1086 ICC (C); *Godlambus* 1114–30 Rams (C); *Godlamb* de Well', de Lenn' 1166 P, 1206 Cur (Nf); Herveus *filius Godlamp* 1221 *ElyA* (C); Roger *Godlamb* 1230 P (Nf); William *Godlomb* 1274 RH (Nf). OE *Gōdlamb*, an original nickname 'good lamb', recorded once before the Conquest. Probably also a nickname.

Goodland: (i) Gilebertus *filius Godland'* 1201 Cur (Nf); Hugo *Godlond* 1279 RH (O). OG *Godland*. (ii) Henry *de Godland* 1379 PTY. 'Dweller by the good land.'

Goodlet: *v.* GOODLAD

Goodley: *v.* GODLEY

Goodliff, Goodliffe: *Gothlif* 1086 ICC (C); *Godelief* 1197 P (K); Maud *Goodleef* 1272 Bart (Lo); John *Godelef* 1296 SRSx; William *Godeloues* 1327 SRWo. OE *Godlēof* (m) or OE **Godlēofu* (f) 'good or god dear', long used in Kent: *Godlefe* (f) 1508 ArchC xxx.

Goodlud: 'This may be a corruption of *Goodlad*, but is more probably for 'good lord'. cf. the legal 'M'lud': John *Godlord* 1332 AD v (Wa); Ralph *Goodlord* 1387 NorwW (Nf).

Goodman: *v.* GODMAN

Goodner: *v.* GOODENOUGH

Goodness: Ranulf *Godnese* 1251 IpmY; Richard *Godnesse* 1327 SRSf. 'Goodness', OE *gōdnes*.

Goodram: *v.* GOODERHAM

Goodread, Goodred, Goodreds, Goodered: *Gudret*, *Godred* 1066 DB (L, Y); *Gudred* de Cnappewelle 1102–7 Rams (C); *Godred* 1200 FFL; Ragemerus *filius Gutred* 1204 P (L); Henry *Guthred* CI214 StGreg (K); John *Godred*, Ysabell *Godrad*, *Godrid* 1279 RH (C); Alexander *Godered'* 1325 FFK; William *Goodrede* 1460 FrY. ON *Guðrøðr*. The surname may also be ME *god rede* 'good counsel'. cf. the Scottish *Meiklereid*, Robert *Smalred* 1176 P (Y), Philip *Lytylred* 1372–6 Gaunt.

รูปที่ 1 แสดงตัวอย่างรายการBase Name และ Variation Name ของชื่ออังกฤษใน พจนานุกรม ของ Reaney and Wilson(1997)

ภาษา PHP

ปัจจุบัน PHP (Hypertext Preprocessor) ได้รับความนิยมเป็นอย่างมาก ในการพัฒนางานบนเว็บที่เรียกว่า Web Programming (ไพศาล, 2544) ซึ่ง PHP ทำงานโดยมีตัวแปลและเอ็กซิคิวต์(Execute) ที่ฝั่งเครื่องเซิร์ฟเวอร์เรียกการทำงานว่าเป็นเซิร์ฟเวอร์ไซด์ (Server Side) ส่วนการทำงานของบราวเซอร์ของผู้ใช้เรียกว่า ไคลเอนต์ (Client Side) โดยการทำงานเริ่มต้นที่ผู้ใช้ส่งข้อความที่ต้องการผ่านเว็บบราวเซอร์ทาง HTTP (HTTP Request) ซึ่งอาจเป็นการกรอกแบบฟอร์มหรือใส่ข้อมูลที่ต้องการ (ข้อมูลเหล่านั้นเป็นเอกสาร PHP) เมื่อเอกสาร PHP เข้ามาถึงเว็บเซิร์ฟเวอร์ก็ถูกส่งไปให้ PHP เพื่อทำหน้าที่แปลคำสั่งแล้วเอ็กซิคิวต์ คำสั่งนั้น หลังจากนั้น PHP สร้างผลลัพธ์ในรูปแบบเอกสาร HTML ส่งกลับไปให้เซิร์ฟเวอร์ เพื่อส่งต่อไปให้บราวเซอร์แสดงผลทางฝั่งผู้ใช้ต่อไป (HTTP Response)

i-678306
วิ QA
278
ก2165
2554

ข้อดีของ PHP มีดังนี้

1. เป็นภาษาแบบ Open Source ซึ่งมีข้อดีดังนี้
   1.1. ไม่มีค่าลิขสิทธิ์
   1.2. Open Source Software หลายตัวมีคุณภาพสูงและได้รับความนิยมจนช่วยกันแก้ปัญหาโดยสามารถซักถามปัญหาผ่านทางอินเทอร์เน็ต ในกรณีที่ตัวเองไม่สามารถแก้ปัญหาเองได้
   1.3. เรียนรู้และดัดแปลงได้ด้วยตัวเอง แก้ไขปัญหาได้ทันที และมีกลุ่มผู้ใช้ร่วมกันช่วยกันแก้ปัญหาโดยสามารถซักถามปัญหาผ่านทางอินเทอร์เน็ตในกรณีที่ตัวเองไม่สามารถแก้ปัญหาเองได้
2. เป็นโปรแกรมที่ฟรี ซึ่งสามารถ Download มาใช้งานได้ฟรี ทั้งตัว Source Code และเอกสารอ้างอิงมาใช้โดยไม่ผิดกฎหมาย
3. สามารถใช้ได้กับหลาย ๆ ระบบปฏิบัติการ เช่น Windows, Unix, Linux
4. PHP สนับสนุนการติดต่อและจัดการกับฐานข้อมูลได้หลายประเภท เช่น MySQL, Oracle และ SQL Server โดยมีการเรียกใช้ผ่านตัวฐานข้อมูลโดยตรง
5. PHP เป็นภาษาที่มีความยืดหยุ่นสูง โดยสามารถแทรกในตำแหน่งใดก็ได้ในแท็กของ HTML โดยเฉพาะความสามารถในการรับส่งข้อมูลจากฟอร์มของ HTML
6. PHP มีประสิทธิภาพการทำงานสูง ทำงานได้รวดเร็ว

ซึ่งความสามารถในการจัดการกับตัวแปรหลาย ๆ ประเภท เช่น เลขจำนวนเต็ม (Integer) เลขทศนิยม (Float) สตริง (String) และอาร์เรย์ (Array) เป็นต้น และสามารถรองรับการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming) และความสามารถในการสร้างกราฟิก

เนื่องจากความสามารถและข้อดีของภาษา PHP มีความยืดหยุ่นในการใช้งานและมีความสามารถในการ Embedded ในแท็กHTML ได้ และ PHP เป็นซอฟท์แวร์แบบเปิดเผย (Open Source) ที่ช่วยลดต้นทุนการดำเนินงาน อีกทั้งยังมีความสามารถที่ติดต่อกับฐานข้อมูลได้หลายโมเดล จึงสะดวกต่อการนำมาใช้งาน

การทบทวนวรรณกรรม/สารสนเทศ ที่เกี่ยวข้อง

Lati และ Randell(1998) ได้ทำการศึกษาประสิทธิภาพและเปรียบเทียบค่าความถูกต้อง (Accuracy measurement) ของ Name Matching Algorithm 4 อัลกอริทึม ซึ่งได้แก่ Soundex,

Guth,K-Approximate และ Metaphoneรวมถึงข้อดีและข้อเสียของแต่ละอัลกอริทึมเพื่อนำไปประยุกต์ใช้ในการสร้างอัลกอริทึมใหม่ ที่ชื่อว่าPhonexโดยใช้ฐานข้อมูลของนามสกุลที่ตีพิมพ์ในหนังสือ Family History Knowledge UK(Park, 1992) ซึ่งมีประมาณ 35,000 รายชื่อในการเปรียบเทียบ จากผลการทดสอบที่ได้ ผู้วิจัยได้นำข้อดีและข้อเสียมาใช้ในการปรับปรุง Name Matching Algorithmโดยคำนึงถึง 3 ด้าน คือด้านความถูกต้องโดยรวมวัดจากร้อยละของ true matches,เวลาที่ใช้เปรียบเทียบ และการออกแบบที่ง่ายต่อการใช้งานทำให้ได้อัลกอริทึมใหม่ ที่มีชื่อว่า Phonex ซึ่งในงานวิจัยนี้ได้วัดประสิทธิอัลกอริทึมพบว่า

1. Guthมีความเร็วในการวัดประสิทธิภาพมากที่สุด
2. Soundexให้ค่าความถูกต้อง (Accuracy) มากที่สุด
3. Phonex ให้ค่า True Matches มากที่สุด และให้ค่าความถูกต้อง (Accuracy) สูงใกล้เคียงกับ Soundex, K-Approximate และ Metaphone

Christen (2006)ได้นำ F-Score มาใช้ในการวัดประสิทธิภาพของ Name Matching Algorithmโดย Name Matching Algorithm ที่ใช้วัดประสิทธิภาพประกอบไปด้วย

1) กลุ่ม Phonetic encoding คือ Soundex, Phonex, Phonix, NYSIIS, Double-Metaphone, Fuzzy soundex

2) กลุ่ม Pattern matching คือ Levenshtein edit distance, Damerau-Levenshtein, Bag distance, Smith-waterman, Longest common sub-string(LCS), n-grams, Positional q-grams, Skip-grams, Compression, Jaro, Winker, Sorted-Winkler, Permuted-Winkler

3) กลุ่ม Combined คือ Editex, Syllable alignment distance

โดยใช้ฐานข้อมูลขนาดใหญ่จำนวน 4 ฐานข้อมูลซึ่งนำมาจากฐานข้อมูลสุขภาพของ Midwives, New south Wales, Australiaดังแสดงในตารางที่ 6โดยฐานข้อมูล3 ฐานข้อมูลแรก คือ Midwives given names (ใช้เฉพาะชื่อ), Midwives surname (ใช้เฉพาะนามสกุล), Midwives full name (เอาชื่อมาต่อด้วยนามสกุลโดยเว้น 1 ช่องว่าง) และฐานข้อมูลสุดท้าย คือ COMPLETE surnames

ตารางที่ 7 แสดงจำนวนระเบียนของฐานข้อมูล

ฐานข้อมูล	Pairs	Singles
Midwives given names	15,233	49,380
Midwives surnames	14,180	79,007
Midwives full names	36,614	339,915
Complete surnames	8,942	13,941

สำหรับอัลกอริทึมประเภท Sound code จากการทดลอง พบว่าName Matching Algorithm ที่ให้ความถูกต้องสูงสุดในฐานข้อมูล 3 ฐานข้อมูลแรก ได้แก่ Phonex และName Matching Algorithm ที่ให้ความถูกต้องสูงสุดในฐานข้อมูลสุดท้าย คือ Phonix

สำหรับอัลกอริทึมประเภทกลุ่มอื่นๆพบว่า LCS-2 ให้ความถูกต้องสูงสุดในข้อมูลชุด Midwives given names ส่วน Jaroให้ความถูกต้องสูงสุดในข้อมูลชุด Midwives surname และ COMPLETE surnamesและในข้อมูลชุด Midwives full nameพบว่า PermWinkให้ความถูกต้องสูงสุด

ด้านความเร็วในการวัดประสิทธิภาพ พบว่า Soundexใช้เวลาน้อยที่สุดในการทำงานในข้อมูลทุกชุดสำหรับอัลกอริทึมประเภท Sound codeส่วนอัลกอริทึมประเภทกลุ่มอื่นๆพบว่า Bag distance ใช้เวลาน้อยที่สุดในการทำงานในข้อมูลทุกชุด

Snae (2007)ได้นำ Name Matching Algorithm ซึ่งได้แก่Guth, Levenshtein, Soundex, Metaphone, NYSIIS, Phonex, ISG, LIG1, LIG2, LIG3 มาเปรียบเทียบโดยใช้เทคนิคการหาค่าความถูกต้อง (Accuracy measurement)ของ Latiและ Randell (1998)โดยใช้ข้อมูลในการทดสอบจากพจนานุกรมนามสกุลภาษาอังกฤษบางส่วนของ Raney and Wilson (1997)เท่านั้น ซึ่งผลที่ได้อาจจะมีความน่าเชื่อถือไม่มากนัก

จากการวัดประสิทธิภาพของ Snea (2007)พบว่ากลุ่มของ Phonex, ISG, LIG1, LIG2และ LIG3 ได้ True matches ค่อนข้างสูง NYSIIS,LIG2, Phonexมีค่าความถูกต้อง(Overall accuracy) สูงและมี True mismatch ค่อนข้างต่ำ (39%, 56%, 63% ตามลำดับ) สำหรับด้านความเร็วพบว่า Metaphone,LIG2, และ LIG3 ใช้เวลามากที่สุด ซึ่งผลการวิจัยพบว่าไม่สามารถใช้ Name Matching Algorithm เดียวในการแก้ปัญหาการจับคู่ชื่อหรือค้นหาชื่อ ดังนั้นวิธีการผสม(Hybrid

approaches)จึงเป็นทางเลือกที่ดีที่สุดส่วนLevenshteinเป็นวิธีการที่เหมาะกับการเปรียบเทียบชื่อในหลายเชื้อชาติ

Panawong and Snae (2010) ได้พัฒนาระบบค้นหาสถานที่ท่องเที่ยวในประเทศไทยโดยใช้ Name Matching Algorithm ในการแก้ปัญหาจากการค้นหาชื่อจังหวัดที่อาจจะมีการเขียนหรือสะกดคล้ายคลึงกัน โดย Name Matching Algorithm ที่นำมาใช้คือ ISG ซึ่งสามารถใช้ได้ทั้งภาษาไทยและภาษาอังกฤษ แต่ข้อเสียของวิธีนี้คือ ไม่สามารถใช้ได้กับชื่อย่อที่ถูกตัดมาจากชื่อเต็มเพราะจะทำให้มีค่า ISG ลดลง

หลักการทำงานของระบบค้นหาสถานที่ท่องเที่ยวในประเทศไทย คือ เมื่อผู้ใช้ต้องการรู้ข้อมูลแหล่งท่องเที่ยวในประเทศไทย ผู้ใช้จะป้อนคำค้นหาจากนั้นระบบจะนำคำที่ค้นหาไปหาคำใกล้เคียงที่อาจจะมีการเขียนหรือสะกดคล้ายคลึงกันหรือที่เกี่ยวข้องโดยใช้ Name Matching Algorithm เพื่อให้ค้นหาได้ครบถ้วนมากที่สุดโดยระบบจะแสดงข้อมูลชื่อที่มีการเขียนได้หลากหลายตามหลักภาษาไทยหรือภาษาอังกฤษ เช่น พิษณุโลก (Phitsanulok), พิษณุโรก (Phitsanurok),พิดโลก (Phitlok),เป็นต้น โดยระบบจะสร้างชื่อหลากหลายแบบอัตโนมัติด้วย Name Matching Algorithm พร้อมทั้งแสดงจำนวนรายการที่พบจากกูเกิ้ล ซึ่งสามารถเลือกดูข้อมูลในแต่ละรายการข้อมูลที่มีชื่อคล้ายคลึงกันกับ พิษณุโลก (ภาษาไทย) หรือ Phitsanulok (ภาษาอังกฤษ) ดังแสดงในตารางที่ 8 และตารางที่ 9 ซึ่งทำให้เกิดความสะดวกและเป็นประโยชน์สำหรับนักท่องเที่ยวชาวต่างประเทศมากยิ่งขึ้น

ตารางที่ 8 แสดงตัวอย่างจำนวนรายการที่ค้นพบโดยกูเกิ้ลของคำค้นหาพิดโลก (ภาษาไทย)

ลำดับ	ชื่อคล้ายคลึง	จำนวนรายการที่พบ (กูเกิ้ล, เมษายน 2553)
1	พิจสะนุโลก	10,100,000
2	พิดโลก	5,770,000
3	พิษณุโลก	5,470,000
4	พิดสะณุโลก	2,290,000
5	พิจสะณุโลก	2,200,000
6	พิสสะนุโลก	1,230,000
7	พิษนุโลก	931,000
8	พิษะณุโลก	110,000
9	พิตสะณุโลก	106,000
10	พิดษะณุโลก	76,200

ตารางที่ 9แสดงตัวอย่างจำนวนรายการที่ค้นพบโดยกูเกิ้ลของคำค้นหาPhitsanulok (ภาษาอักกฤษ)

ลำดับ	ชื่อคล้ายคลึง	จำนวนรายการที่พบ (กูเกิ้ล, เมษายน 2553)
1	phitsanulok	1,240,000
2	phisanulok	603,000
3	pitsanulok	117,000
4	pisanulok	17,200
5	phitsanurok	1,910
6	phisanuklok	105
7	phitzanulok	53
8	phitsanoolok	6
9	phitsanoklok	3
10	phisanooklok	1

ในงานวิจัยนี้ได้นำName Matching Algorithm 12 อัลกอริทึมมาพัฒนาโปรแกรมการเปรียบเทียบชื่อสองชื่อว่ามีความคล้ายคลึงกันหรือไม่เพื่อใช้ในการวัดประสิทธิภาพและเปรียบเทียบ Name Matching Algorithm โดยใช้วิธีการหาค่า True Match, ค่า True Mismatch และค่าความถูกต้อง (Overall accuracy) ตามแนวคิดของ Laitและ Randell (1998)วิธีการ F-measure ตามแนวคิดของ Christen (2006), วิธีการจัดกลุ่ม และ Clustering ตามแนวคิดของ Snae (2006) และเวลาที่ใช้ในการประมวลผล Name Matching Algorithm โดยใช้ฐานข้อมูลชื่อทั้งหมดของA Dictionary of English Surnames ของ Reaney andWilson (1997)

บทที่ 3

วิธีดำเนินงานวิจัย

การวิจัยครั้งนี้เป็นการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมโดยมีหัวข้อในการดำเนินงานวิจัยดังนี้

- อุปกรณ์ที่ใช้ในการดำเนินงานวิจัย
- ขั้นตอนการดำเนินงานวิจัย

1. อุปกรณ์ที่ใช้ในการดำเนินงานวิจัย

อุปกรณ์ที่ใช้ในการดำเนินงานวิจัย ประกอบด้วยเครื่องคอมพิวเตอร์จำนวน 1 เครื่อง ส่วนประกอบที่สำคัญ คือ

อุปกรณ์ด้านฮาร์ดแวร์ ได้แก่

- หน่วยประมวลผลกลาง –Intel® Dual Core 1.86 GHz
- หน่วยความจำหลัก 1 GB

อุปกรณ์ด้านซอร์ฟแวร์ ได้แก่

- ระบบปฏิบัติการ Microsoft Windows XP SP3
- โปรแกรม AppServ2.6.0
- โปรแกรม Adobe Dreamweaver CS5
- โปรแกรม Internet Explorer

2. ขั้นตอนการดำเนินงานวิจัย

งานวิจัยนี้มีขั้นตอนการดำเนินการวิจัย ดังนี้

3.1 ขั้นตอนการศึกษาและวิเคราะห์หลักการที่นำไปใช้ในการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึม

ผู้วิจัยได้ศึกษาแนวคิด ทฤษฎี หลักการ ตลอดจนเอกสารงานวิจัยที่เกี่ยวข้องในประเทศและต่างประเทศ ตามที่ได้เสนอในบทที่ 2 เพื่อเป็นพื้นฐานความรู้สำหรับการทำวิจัยเกี่ยวกับการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมโดยสามารถแบ่งออกเป็น 2 กลุ่มคือ

1) กลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง

เนมแมทชิ่งอัลกอริทึมในกลุ่มนี้เป็นการเปรียบเทียบระหว่างรหัสเสียงของชื่อ โดยการนำรหัสเสียงของชื่อ 2 ชื่อมาเปรียบเทียบกันโดยตรง ถ้าเหมือนกันแสดงว่าชื่อทั้ง 2 ชื่อ Match กัน แต่ถ้าไม่เหมือนกันแสดงว่าชื่อทั้ง 2 ชื่อ No Match กัน ซึ่งเนมแมทชิ่งอัลกอริทึมในกลุ่มนี้มี 6 อัลกอริทึมได้แก่ Soundex, Soundex2, Phonex, Metaphone, Simplex และ NYSIIS

2) กลุ่มที่มีการเปรียบเทียบกับค่าThreshold

การเปรียบเทียบกับค่า Thresholdเป็นวิธีการนำชื่อ 2 ชื่อมาหาค่าตามวิธีการของแต่ละอัลกอริทึม แล้วนำค่านั้นไปเปรียบเทียบกับค่าThresholdจึงจะสามารถสรุปได้ว่าชื่อ 2 ชื่อนั้น Match กันหรือไม่ เนมแมทชิ่งอัลกอริทึมในกลุ่มนี้มี 6 อัลกอริทึมได้แก่ Guth, Levenshtein, ISG, LIG1, LIG2 และ LIG3 ซึ่งการกำหนดค่าThreshold มีผลต่อการเปรียบเทียบชื่อและการวัดประสิทธิภาพเป็นอย่างมาก ถ้ากำหนดไม่เหมาะสมอาจทำให้การเปรียบเทียบชื่อไม่มีความน่าเชื่อถือและการวัดประสิทธิภาพมีผลคาดเคลื่อนได้ดังนั้นผู้วิจัยจึงนำค่า F-Score มาใช้ในเปรียบเทียบเพื่อหาค่าที่เหมาะสมที่สุดในการกำหนดค่าThresholdซึ่งแต่ละอัลกอริทึมจะมีการเปรียบเทียบค่าThreshold ที่แตกต่างกัน

- Levenshteinเป็นการนำค่า Levenshtein Distance (LD)มา เปรียบเทียบค่า Threshold ถ้า LD น้อยกว่าหรือเท่ากับค่าThreshold จะให้ผล Match ดังนั้นค่าThreshold ของ Levenshtein จะเป็นร้อยละของค่าเฉลี่ยของจำนวนตัวอักษรของชื่อ 2 ชื่อ
- Guthเป็นวิธีการนับจำนวนตัวอักษรที่เหมือนกันเปรียบเทียบค่า Thresholdถ้าจำนวนตัวอักษรที่เหมือนกันมากกว่าหรือเท่ากับค่าThreshold จะให้ผล Match ดังนั้น ค่าThreshold ของ Guthจะเป็นร้อยละของค่าเฉลี่ยของจำนวนตัวอักษรของชื่อ 2 ชื่อ
- สำหรับ ISG, LIG1, LIG2 และ LIG3 มีสูตรการค่าหาของแต่ละอัลกอริทึม โดยค่านี้แสดงถึงความคล้ายคลึงกันของชื่อ ยิ่งค่านี้มากจะแสดงถึงความคล้ายคลึงที่มากด้วยดังนั้นการกำหนดค่าThreshold เพื่อหาค่าThreshold ที่ดีที่สุดจะกำหนดที่ 0.6, 0.7, 0.8และ 0.9โดยผลการเปรียบเทียบสรุปได้ว่าค่า Threshold ที่เหมาะสมที่สุดของISG, LIG1, LIG2 และ LIG3 มีค่าดังต่อไปนี้ 0.8, 0.7, 0.8 และ 0.9 ตามลำดับ(จากผลการทดสอบในบทที่ 4หัวข้อ1)

3.2 ขั้นตอนการพัฒนาเนมแมทชิ่งอัลกอริทึมและการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

ผู้วิจัยได้ทำการพัฒนาเนมแมทชิ่งอัลกอริทึมและการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมโดยทำการพัฒนาเนมแมทชิ่งอัลกอริทึมทั้งหมด 12 อัลกอริทึม ซึ่งพัฒนาด้วยภาษา PHP โดยในงานวิจันนี้ได้ใช้การวัดประสิทธิภาพทั้งหมด 3 วิธีการ ดังนี้

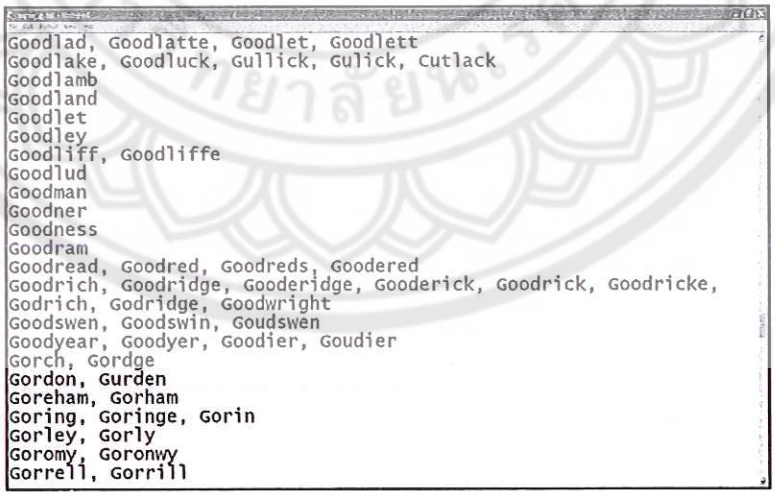
1) การวัดประสิทธิภาพตามแนวคิดของ Laitและ Randell (1998)โดยวิธีการนี้สามารถวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมได้ทั้ง 2 กลุ่มรวม12 อัลกอริทึม

2) การวัดประสิทธิภาพด้วยวิธีการ F-Measureโดยวิธีการนี้สามารถวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมได้ทั้ง 2 กลุ่มรวม 12 อัลกอริทึม

3) การวัดประสิทธิภาพด้วยเทคนิค Clusteringโดยวิธีการนี้สามารถวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมได้เพียงกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง(กลุ่มที่ 1)เท่านั้นรวม 6 อัลกอริทึม

3.3 การสร้างข้อมูลรายชื่อของพจนานุกรมนามสกุลภาษาอังกฤษเพื่อวัดประสิทธิภาพ

พจนานุกรมนามสกุลภาษาอังกฤษ (A Dictionary of English Surname) ของ Raney และ Wilson (1997) ที่มีข้อมูลเกี่ยวกับนามสกุล โดยรายการชื่อในพจนานุกรมแต่ละรายการจะประกอบไปด้วย Base name แล้วตามด้วยVariation name ซึ่งในงานวิจัยนี้ได้ทำสร้างไฟล์รายชื่อ โดยในแต่ละแถว ประกอบด้วย ชื่อแรกเป็น Base Name และตามด้วย Variation Name เพื่อใช้ในการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมดังแสดงในภาพ 2



ภาพ 2 ตัวอย่างไฟล์รายการชื่อของพจนานุกรมนามสกุลภาษาอังกฤษ

3.4 ขั้นตอนการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell

ขั้นตอนการวัดประสิทธิภาพและเปรียบเทียบ ตามแนวคิดของ Laitและ Randell (1998)นี้เป็นการคำนวณหาค่า True Matches เป็นค่าร้อยละของจำนวนชื่อที่ Match ได้ถูกต้องเทียบกับจำนวนชื่อที่ควร Match ทั้งหมดค่า True Mismatches เป็นค่าร้อยละของจำนวนชื่อที่ Match ได้ไม่ถูกต้องเทียบกับจำนวนชื่อที่ Match ทั้งหมดและค่า Overall Accuracy หรือค่าความถูกต้อง เป็นค่าร้อยละของจำนวนชื่อที่ Match ได้ถูกต้องเทียบกับจำนวนชื่อที่ Match ทั้งหมด

อย่างไรก็ตามการวัดและเปรียบเทียบประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell(1998)สามารถวัดและเปรียบเทียบประสิทธิภาพได้ทั้ง2 กลุ่ม ดังต่อไปนี้

กลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง มีตัวแปรที่เกี่ยวข้อง คือ

- ค่า kคือ จำนวน Variation nameกับ Basename อื่นที่ Match กับ Basename
- ค่า a คือ จำนวน Variation name ที่ Match กับ Basename

ขั้นตอนในการวัดประสิทธิภาพ มีดังต่อไปนี้

1) กำหนดค่า k=0, a=0 โดยที่
2) กำหนด n=1 และ num1=จำนวน Base name ทั้งหมดในรายชื่อ R&W
3) อ่าน Base name(n) จากรายชื่อ R&W
4) กำหนด m=1 และ num2=จำนวนของ Variation name ของ Base name(n)
5) อ่าน Variation name(m)ของ Base name(n) จากรายชื่อ R&W
6) กำหนด S1=รหัสเสียงของ Base name(n) และ S2=รหัสเสียงของ Variation name(m)
7) พิจารณา S1=S2 หรือไม่
   - 7.1) ถ้า ใช่ :k=k+1 , a=a+1
   - 7.2) ถ้า ไม่ใช่ : ทำข้อ 8) ต่อไป
8) พิจารณา m=num2 หรือไม่
   - 8.1) ถ้า ใช่ : ทำข้อ 9) ต่อไป
   - 8.2) ถ้า ไม่ใช่ : m=m+1 แล้วกลับไปทำข้อ 5)

9) กำหนด i=1 และ num3=จำนวนของ Base name อื่น

10) อ่าน Base name อื่น(i) จากรายชื่อ R&W

11) กำหนด S3=รหัสเสียงของ Base name อื่น(i)

12) พิจารณา S1=S3 หรือไม่

12.1) ถ้า ใช่ :k=k+1

12.2) ถ้า ไม่ใช่ : ทำข้อ 13) ต่อไป

13) พิจารณา i=num3 หรือไม่

13.1) ถ้า ใช่ :ทำข้อ 14) ต่อไป

13.2) ถ้า ไม่ใช่ : i=i+1 แล้วกลับไปทำข้อ 10)

14) พิจารณา n=num1 หรือไม่

14.1) ถ้า ใช่ :ทำข้อ 15) ต่อไป

14.2) ถ้า ไม่ใช่ : คำนวณหาค่า True Matches, ค่า True Mismatchesและค่า Overall Accuracyของ Base name(n) ดังแสดงในบทที่ 2 ข้อ 2.1และ n=n+1, k=0, a=0แล้วกลับไปทำข้อ 3)

15) คำนวณหาค่า True Matches, ค่า True Mismatchesและค่า Overall Accuracy ของเนมแมทชิ่งอัลกอริทึมดังแสดงในบทที่ 2 หัวข้อ 2.1

จากขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell(1998) กลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง ผู้วิจัยจะนำเสนอตัวอย่างโดยใช้ Soundex ดังตารางที่ 10

ตารางที่10 แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randellของกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง (Soundex)

Base	Soundex	R&W	Soundex	Result

Name	Code		Code	(Match/No Match)	
		v1 Goodlatte	G343	Match	k1,a1
		v2 Goodlet	G343	Match	k2, a2
		v3 Goodlett	G343	Match	k3, a3
		Goodlake	G342	No Match	
		Goodlamb	G345	No Match	
		Goodland	G345	No Match	
		Goodlet	G343	Match	k4
Goodladk	G343	Goodley	G340	No Match	
		Goodliff	G341	No Match	
		Goodlud	G343	Match	k5
		Goodman	G355	No Match	
		Goodner	G356	No Match	
		Goodness	G352	No Match	
		Goodram	G365	No Match	
		Goodread	G363	No Match	

หมายเหตุ: ส่วนสีเทา คือ Variation name เดียวกับ Base name

ส่วนสีขาว คือ Base name อื่น

จากตารางที่ 15 สามารถคำนวณหาค่าต่าง ๆ ได้ดังต่อไปนี้

- หาค่า k ได้จากจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ รายชื่อ R&W ซึ่งจากตารางที่ 10 k เท่ากับ 5 (Goodlatte, Goodlet, Goodlett, Goodletและ Goodlud)
- หาค่า a โดยการนำจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ Variation name เดียวกันซึ่งจากตารางที่ 10 a เท่ากับ 3 (Goodlatte, GoodletและGoodlett)
- **หาค่า v ได้จากจำนวน Variation name ในชื่อเดียวกันซึ่งจากตารางที่ 15ค่า v เท่ากับ 3 (Goodlatte, Goodletและ Goodlett)**

- คำนวณหาค่า True Match, ค่า True Mismatch และค่า Overall accuracy ของ Base name (สูตรในบทที่ 2หัวข้อ 2.1) ยกตัวอย่างจากตารางที่ 16 Goodladสามารถหาค่าต่าง ๆ ได้ดังนี้

  - True Matches : 3/3*100 = 100 %
  - True Mismatches : (5-3)/5*100 = 40%
  - Overall Accuracy : 3/5*100 = 60%

โดยทำการหาค่า True Match, ค่า True Mismatch และค่า Overall accuracyของทุก Base name

- คำนวณหาค่า True Match, ค่า True Mismatch และค่า Overall accuracyของเนมแมทชิ่งอัลกอริทึม(สูตรในบทที่ 2หัวข้อ 2.1)จากตารางที่ 16 สามารถหาค่าต่าง ๆ ได้ดังนี้

  - True Matches : (100+25+100+100)/4 = 81.25%
  - True Mismatches : (40+0+0+0)/4 = 10%
  - Overall Accuracy : (60+100+100+100)/4 = 90%

กลุ่มที่มีการเปรียบเทียบกับค่าThreshold โดยมีตัวแปรที่เกี่ยวข้อง คือ

- ค่า kคือ จำนวน Variation nameกับ Basenameอื่นที่ Match กับ Basename
- ค่า a คือ จำนวน Variation name ที่ Match กับ Basename

โดยมีขั้นตอนในการวัดประสิทธิภาพดังต่อไปนี้

1) กำหนดค่า k=0 และ a=0 โดยที่
2) กำหนด n=1 และ num1=จำนวน Base name ทั้งหมดในรายชื่อ R&W
3) อ่าน Base name(n) จากรายชื่อ R&W
4) กำหนด m=1 และ num2=จำนวนของ Variation name ของ Base name(n)
5) อ่าน Variation name(m)ของ Base name(n) จากรายชื่อ R&W
6) กำหนด S1=Name Matching Score ของ Base name(n) กับ Variation name(m)และ T=ค่า Threshold
7) พิจารณา S1>=S2 หรือไม่ (Levenshtein : S1<=S2)

7.1) ถ้า ใช่ :k=k+1 , a=a+1

7.2) ถ้า ไม่ใช่ :ทำข้อ 8) ต่อไป

8) พิจารณา m=num2 หรือไม่

8.1) ถ้า ใช่ : ทำข้อ 9) ต่อไป

8.2) ถ้า ไม่ใช่ : m=m+1 แล้วกลับไปทำข้อ 5)

9) กำหนด i=1 และ num3=จำนวนของ Base name อื่น

10) อ่าน Base name อื่น(i) จากรายชื่อ R&W

11) กำหนด S1=Name Matching Score ของ Base name(n) กับ Base name อื่น(i)และ T=ค่า Threshold

12) พิจารณา S1>=S3 หรือไม่ (Levenshtein : S1<=S2)

12.1) ถ้า ใช่ :k=k+1

12.2) ถ้า ไม่ใช่ : ทำข้อ 13) ต่อไป

13) พิจารณา i=num3 หรือไม่

13.1) ถ้า ใช่ :ทำข้อ 14) ต่อไป

13.2) ถ้า ไม่ใช่ : i=i+1 แล้วกลับไปทำข้อ 10)

14) พิจารณา n=num1 หรือไม่

14.1) ถ้า ใช่ :ทำข้อ 15) ต่อไป

14.2) ถ้า ไม่ใช่ : คำนวณหาค่า True Matches, ค่า True Mismatchesและค่า Overall Accuracyของ Base name(n) ดังแสดงในบทที่ 2 ข้อ 2.2 และ n=n+1, k=0, a=0แล้วกลับไปทำข้อ 3)

15) คำนวณหาค่า True Matches, ค่า True Mismatchesและค่า Overall Accuracy ของเนมแมทชิ่งอัลกอริทึมดังแสดงในบทที่ 2 ข้อ 2.2

จากขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell(1998)กลุ่มที่มีการเปรียบเทียบกับค่าThresholdผู้วิจัยจะนำเสนอตัวอย่างโดยใช้ ISG ดังตารางที่ 11

ตารางที่11 แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randellของกลุ่มที่มีการเปรียบเทียบกับค่าThreshold (ISG)

Base Name	R&W	ISG (Threshold=0.6)	Result (Match/No Match)	
Goodlad	Goodlatte	0.6	Match	k1 ,a1
	Goodlet	0.556	No Match	
	Goodlett	0.5	No Match	
	Goodlake	0.667	Match	k2
	Goodlamb	0.667	Match	k3
	Goodland	0.875	Match	k4
	Goodlet	0.556	No Match	
	Goodley	0.556	No Match	
	Goodliff	0.5	No Match	
	Goodlud	0.75	Match	k5
	Goodman	0.556	No Match	
	Goodner	0.4	No Match	
	Goodness	0.364	No Match	
	Goodram	0.556	No Match	
	Goodread	0.667	Match	k6

หมายเหตุ: ส่วนสีเทา คือ Variation name เดียวกับ Base name

ส่วนสีขาว คือ Base name อื่น

จากตารางที่ 11 สามารถคำนวณหาค่าต่าง ๆ ได้ดังต่อไปนี้

- หาค่า k ได้จากจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ รายชื่อ R&W ซึ่งจากตารางที่ 11k เท่ากับ 6 (Goodlatte, Goodlake, Goodlamb, Goodland, Goodludและ Goodread)

- หาค่า a โดยการนำจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ Variation name เดียวกันซึ่งจากตารางที่ 17aเท่ากับ 1 (a1)
- หาค่า v ได้จากจำนวน Variation name ในชื่อเดียวกันซึ่งจากตารางที่17ค่าv เท่ากับ 3 (v1, v2 และ v3)
- คำนวณหาค่า True Match, ค่า True Mismatch และค่า Overall accuracyของ Base name(สูตรในบทที่ 2หัวข้อ 2.1) ยกตัวอย่างจากตารางที่ 18Goodladสามารถหาค่าต่าง ๆ ได้ดังนี้

  - True Matches : 1/3*100 = 33.333 %
  - True Mismatches : (6-1)/6*100 = 83.333%
  - Overall Accuracy : 1/6*100 = 16.667%

  โดยทำการหาค่า True Match, ค่า True Mismatch และค่า Overall accuracyของทุก Base name

- คำนวณหาค่า True Match, ค่า True Mismatch และค่า Overall accuracyของเนมแมทชิ่งอัลกอริทึม(สูตรในบทที่ 2หัวข้อ 2.1) จากตารางที่ 18 สามารถหาค่าต่าง ๆ ได้ดังนี้
  - True Matches : (33.333+25+100+100)/4 = 64.583%
  - True Mismatches : (83.333+85.714+0+62.5)/4 = 57.887%
  - Overall Accuracy : (16.667+14.286+100+37.5)/4 = 42.113%

3.5 ขั้นตอนการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการF-Measure

การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-Measure สามารถแบ่งการวัดประสิทธิภาพออกเป็น 2 กลุ่ม ดังที่กล่าวในบทที่ 3 ข้อ 3.1 โดยกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง มีตัวแปรที่เกี่ยวข้อง คือ

- ค่า TP (True Positive) คือ จำนวน Variation name ที่ Match กับ Basename
- ค่า FP (False Positive) คือ จำนวน Basename อื่น ที่ Match กับ Basename
- ค่า FN (False Negative) คือ จำนวน Variation name ที่ No Match กับ Basename

โดยมีขั้นตอนในการวัดประสิทธิภาพดังต่อไปนี้

1) กำหนดค่า TP=0, FP=0และ FN=0โดยที่

2) กำหนด n=1 และ num1=จำนวน Basename ทั้งหมดในรายชื่อR&W

3) อ่าน Base name(n)จากรายชื่อ R&W

4) กำหนด m=1 และ num2=จำนวนของ Variationname ของ Base name(n)

5) อ่าน Variation name(m)ของ Base name(n) จากรายชื่อ R&W

6) กำหนด S1=รหัสเสียงของBase name(n)และ S2=รหัสเสียงของ Variation name(m)

7) พิจารณา S1=S2 หรือไม่

   7.3) ถ้าใช่:TP=TP+1

   7.4) ถ้า ไม่ใช่ :FN=FN+1

8) พิจารณา m=num2 หรือไม่

   8.1) ถ้า ใช่ : ทำข้อ 9) ต่อไป

   8.2) ถ้า ไม่ใช่ : m=m+1 แล้วกลับไปทำข้อ 5)

9) กำหนด i=1 และ num3=จำนวนของ Base name อื่น

10) อ่าน Base name อื่น(i) จากรายชื่อ R&W

11) กำหนด S3=รหัสเสียงของ Base nameอื่น(i)

12) พิจารณา S1=S3 หรือไม่

   12.3) ถ้า ใช่ :FP=FP+1

   12.4) ถ้า ไม่ใช่ : ทำข้อ 13) ต่อไป

13) พิจารณา i=num3 หรือไม่

   13.3) ถ้า ใช่ :ทำข้อ 14) ต่อไป

   13.4) ถ้า ไม่ใช่ : i=i+1 แล้วกลับไปทำข้อ 10)

14) พิจารณา n=num1 หรือไม่

   14.3) ถ้า ใช่ :ทำข้อ 15) ต่อไป

   14.4) ถ้า ไม่ใช่ : คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของ Base name(n)ดังแสดงในบทที่ 2 ข้อ 2.2 และ n=n+1, TP=0, FP=0, FN=0แล้วกลับไปทำข้อ 3)

15) คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของเนมแมทชิ่งอัลกอริทึม ดังแสดงในบทที่ 2 ข้อ 2.2

จากขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure กลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง ผู้วิจัยจะนำเสนอตัวอย่างโดยใช้ Soundex ดังตารางที่ 12

ตารางที่12 แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการF-measure ของกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง (Soundex)

Base Name	Soundex Code	R&W	Soundex Code	Result (Match/No Match)	
Goodlake	G342	Goodluck	G342	Match	TP1
		Gullick	G420	No Match	FN1
		Gulick	G420	No Match	FN2
		Cutlack	C342	No Match	FN3
		Goodlad	G343	No Match	FP=0
		Goodlamb	G345	No Match	
		Goodland	G345	No Match	
		Goodlet	G343	No Match	
		Goodley	G340	No Match	
		Goodliff	G341	No Match	
		Goodlud	G343	No Match	
		Goodman	G355	No Match	
		Goodner	G356	No Match	
		Goodness	G352	No Match	
		Goodram	G365	No Match	
		Goodread	G363	No Match	

หมายเหตุ: ส่วนสีเทา คือ Variation name เดียวกับ Base name

ส่วนสีขาว คือ Base name อื่น

จากตารางที่ 12 สามารถคำนวณหาค่าต่าง ๆ ได้ดังต่อไปนี้

- หาค่า TPได้จากการนับชื่อที่ Match ในการเปรียบเทียบ Soundex Code ของ Base name กับ Soundex Code ของ Variation name ซึ่งจากตารางที่ 19TPเท่ากับ 1 (Goodluck)
- หาค่า FPได้จากการนับชื่อที่ Match ในการเปรียบเทียบ Soundexของ Base name กับ Soundexของ Base name อื่นซึ่งจากตารางที่ 19FPเท่ากับ 0เนื่องจากไม่มีค่า Match ใน Base name อื่น
- หาค่า FNได้จากการนับชื่อที่No match ในการเปรียบเทียบ Soundex Code ของ Base name กับ Soundex Code ของ Variation nameซึ่งจากตารางที่ 19ค่าFNเท่ากับ 3 (Gullick, Gulickและ Cutlack)
- คำนวณหาค่า Precision, ค่า Recallและค่า F-Scoreของ Base name(สูตรในบทที่ 2 หัวข้อ 2.2) ยกตัวอย่างจากตารางที่ 20 Goodladสามารถหาค่าต่าง ๆ ได้ดังนี้
  - Precision : 3/(3+2)*100 = 60 %
  - Recall : 3/(3+0)*100 = 100%
  - F-Score : 2*(60*100)/(60+100) = 75%

  โดยทำการหาค่า Precision, ค่า Recallและค่า F-Score ของทุก Base name
- คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของเนมแมทชิ่งอัลกอริทึม(สูตรในบทที่ 2 ข้อ 2.2)จากตารางที่ 20 สามารถหาค่าต่าง ๆ ได้ดังนี้
  - Precision : (60+100+100+100) = 90 %
  - Recall : (100+25+100+100) = 81.25%
  - F-Score : (75+40+100+100)/4 = 78.75%

สำหรับกลุ่มที่มีการเปรียบเทียบกับค่า Threshold โดยมีตัวแปรที่เกี่ยวข้อง คือ

- ค่า TP (True Positive) คือ จำนวน Variation name ที่ Match กับ Basename
- ค่า FP (False Positive) คือ จำนวน Basename อื่น ที่ Match กับ Basename
- ค่า FN (False Negative) คือ จำนวน Variation name ที่ No Match กับ Basename

มีขั้นตอนในการวัดประสิทธิภาพดังต่อไปนี้

1) กำหนดค่า TP=0, FP=0 และ FN=0 โดยที่

2) กำหนด n=1 และ num1=จำนวน Base name ทั้งหมดในรายชื่อ R&W

3) อ่าน Base name(n) จากรายชื่อ R&W

4) กำหนด m=1 และ num2=จำนวนของ Variation name ของ Base name(n)

5) อ่าน Variation name(m)ของ Base name(n) จากรายชื่อ R&W

6) กำหนด S1=Name Matching Score ของ Base name(n) กับ Variation name(m)และ T=ค่า Threshold

7) พิจารณา S1>=S2 หรือไม่ (Levenshtein : S1<=S2)

   7.1) ถ้า ใช่ :TP=TP+1

   7.2) ถ้า ไม่ใช่ :FN=FN+1

8) พิจารณา m=num2 หรือไม่

   8.1) ถ้า ใช่ : ทำข้อ 9) ต่อไป

   8.2) ถ้า ไม่ใช่ : m=m+1 แล้วกลับไปทำข้อ 5)

9) กำหนด i=1 และ num3=จำนวนของ Base name อื่น

10) อ่าน Base name อื่น(i) จากรายชื่อ R&W

11) กำหนด S1=Name Matching Score ของ Base name(n) กับ Base name อื่น(i)และ T=ค่า Threshold

12) พิจารณา S1>=S3 หรือไม่ (Levenshtein : S1<=S2)

   12.1) ถ้า ใช่ :FP=FP+1

   12.2) ถ้า ไม่ใช่ : ทำข้อ 13) ต่อไป

13) พิจารณา i=num3 หรือไม่

   13.1) ถ้า ใช่ :ทำข้อ 14) ต่อไป

   13.2) ถ้า ไม่ใช่ : i=i+1 แล้วกลับไปทำข้อ 10)

14) พิจารณา n=num1 หรือไม่

   14.1) ถ้า ใช่ :ทำข้อ 15) ต่อไป

14.2) ถ้า ไม่ใช่ : คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของ Base name(n) ดังแสดงในบทที่ 2 ข้อ 2.2 และ n=n+1, TP=0, FP=0, FN=0แล้วกลับไปทำข้อ 3)

15) คำนวณหาค่า Precision, ค่า Recall และค่า F-Score ของเนมแมทชิ่งอัลกอริทึม ดังแสดงในบทที่ 2 ข้อ 2.2

จากขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-Measureกลุ่มที่มีการเปรียบเทียบกับค่าThreshold ผู้วิจัยจะนำเสนอตัวอย่างโดยใช้ ISG ดังตารางที่ 21

ตารางที่21แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure ของกลุ่มที่มีการเปรียบเทียบกับค่าThreshold (ISG)

Base Name	R&W	ISG Score (Threshold=0.6)	Result (Match/No Match)	
	Goodluck	0.6	Match	TP1
	Gullick	0.25	No Match	FN1
	Gulick	0.273	No Match	FN2
	Cullack	0.25	No Match	FN3
	Goodlad	0.6	Match	FP1
	Goodlamb	0.6	Match	FP2
	Goodland	0.6	Match	FP3
Goodlake	Goodlet	0.667	Match	FP4
	Goodley	0.667	Match	FP5
	Goodliff	0.455	No Match	
	Goodlud	0.5	No Match	
	Goodman	0.5	No Match	
	Goodner	0.5	No Match	
	Goodness	0.455	No Match	
	Goodram	0.5	No Match	

	Goodread	0.6	Match	FP6

หมายเหตุ: **ส่วนสีเทา คือ Variation name เดียวกับ Base name**

ส่วนสีขาว คือ Base name อื่น

จากตารางที่ 21 สามารถคำนวณหาค่าต่าง ๆ ได้ดังต่อไปนี้

- หาค่า TP ได้จากจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ Variation name เดียวกัน ซึ่งจากตารางที่ 21TP เท่ากับ 1 (Goodluck)
- หาค่า FP โดยการนำจำนวนชื่อที่ Match ในการเปรียบเทียบ Base name กับ Base name อื่นซึ่งจากตารางที่ 21FP เท่ากับ 6(Goodlad, Goodlamb, Goodland, Goodlet, Goodley และ Goodread)
- หาค่า FN ได้จากจำนวนNo match แต่ควร Match ซึ่งจากตารางที่ 21ค่าFNเท่ากับ 3 (Gullick, Gulickและ Cutlack)
- คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของ Base name(สูตรในบทที่ 2 ข้อ 2.2) ยกตัวอย่างจากตารางที่ 22 Goodladสามารถหาค่าต่าง ๆ ได้ดังนี้
  - Precision : 1/(1+5)*100 = 16.667 %
  - Recall : 1/(1+2)*100 = 33.333%
  - F-Score : 2*(16.667*33.333)/(16.667+33.333) = 22.222%

  โดยทำการหาค่า Precision, ค่า Recall และค่า F-Score ของทุก Base name
- คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของ R&W(สูตรในบทที่ 2หัวข้อ 2.2) จากตารางที่ 22 สามารถหาค่าต่าง ๆ ได้ดังนี้
  - Precision : (16.667+14.286+100+37.5) = 42.113 %
  - Recall : (33.333+25+100+100) = 64.583%
  - F-Score : (22.222+18.182+100+54.545)/4 = 48.737%

3.6 ขั้นตอนการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยเทคนิค Clustering

การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering สามารถแบ่งการวัดประสิทธิภาพออกเป็น 2 แบบคือ Included single clusterและIgnored single cluster ดังที่กล่าวในบทที่ 2ข้อ 2.3มีขั้นตอนในการวัดประสิทธิภาพดังต่อไปนี้

1) สร้างรายชื่อทั้งหมดใน R&Wดังแสดงในตารางที่ 23 คอลัมน์ที่ 1 (R&W)

2) สร้าง Group1 โดยBase name(n) และ Variation name ของ Base name(n) อยู่ในกลุ่มเดียวกัน ดังแสดงในตารางที่ 23 คอลัมน์ที่ 2(Group1)

3) สร้าง Soundcode โดยแปลงรายชื่อทั้งหมดใน R&Wเป็นรหัสเสียงดังแสดงในตารางที่ 23 คอลัมน์ที่ 3

4) เรียงลำดับ Soundcodeดังแสดงในตารางที่ 23 คอลัมน์ที่ 4

5) สร้าง Group2 โดย Soundcodeเดียวกัน อยู่ในกลุ่มเดียวกันดังแสดงในตารางที่ 23 คอลัมน์ที่ 5

6) สร้าง Cluster โดยการหาความถี่ของกลุ่มเดียวกันที่มากที่สุดระหว่าง Group1 กับ Group2 โดยเริ่มจาก Cluster 1 และทำซ้ำจนครบทุกชื่อดังแสดงในตารางที่ 23 คอลัมน์ที่ 6

   a. n=ค่าความถี่ของกลุ่มเดียวกันที่มากที่สุดระหว่าง Group1 กับ Group2

   b. k=ค่าความถี่ของกลุ่มเดียวกันที่มากเป็นลำดับ 2 ระหว่าง Group1 กับ Group2

   c. คำนวณหาค่า chจากสูตรในบทที่ 2 หัวข้อ 2.3

7) คำนวณหาค่าประสิทธิภาพของเนมแมทชิ่งอัลกอริทึมได้จากสูตรในบทที่2 หัวข้อ 2.3

จากขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClusteringผู้วิจัยจึงเสนอตัวอย่างโดยใช้ Soundex ดังแสดงในตารางที่ 13

ตารางที่ 13 แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering ของ Soundex

R&W	Group1		Soundex	Soundex (Sort)	Group2		Cluster	n	k	ch	lg
Goodlad	1	n1	G343	C342	1						
Goodlatte	1	n2	G343	G340	2						
Goodlet	1	n3	G343	G341	3	k1	1	4	2	75	0
Goodlett	1	n4	G343	G341	3	k2					
Goodlake	2		G342	G342	4						
Goodluck	2		G342	G342	4						
Gullick	2		G420	G343	5		2	5	3	80	0
Gulick	2		G420	G343	5						
Cutlack	2		C342	G343	5						
Goodlamb	3		G345	G343	5						
Goodland	4		G345	G343	5		3	3	1	66.67	0
Goodlet	5		G343	G343	5						
Goodley	6		G340	G345	6		4	2	1	75	0
Goodliff	7		G341	G345	6						
Goodliffe	7		G341	G350	7		5	1	1	100	100
Goodlud	8		G343	G352	8		6	1	1	100	100
Goodman	9		G350	G356	9		7	1	1	100	100
Goodner	10		G356	G363	10						
Goodness	11		G352	G363	10		8	3	1	66.67	0
Goodram	12		G365	G363	10						
Goodread	13		G363	G363	10						
Goodred	13		G363	G365	11						
Goodreds	13		G363	G420	12		9	3	2	83.34	0
Goodered	13		G363	G420	12						
										742.5	300

จากตารางที่ 13 สามารถคำนวณค่าค่าต่าง ๆ ดังต่อไปนี้

- หาค่า nได้จากค่าความถี่ของกลุ่มเดียวกันที่มากที่สุดระหว่าง Group1 กับ Group2ซึ่งจากตารางที่ 23 ค่า n ของ Cluster ที่ 1 เท่ากับ4 (เลข 1 ของGroup1)
- หาค่า k ได้จากค่าความถี่ของกลุ่มเดียวกันที่มากเป็นลำดับ 2 ระหว่าง Group1 กับ Group2ซึ่งจากตารางที่ 23 ค่า k ของ Cluster ที่ 1 เท่ากับ 2 (เลข 3 ของ Group2)

ตารางที่ 14 แสดงตัวอย่างผลของการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิค Clusteringของ Soundex

All cluster(I=9)	n	k	2n	ch(i)(%)	ig(i)(%)
Cluster 1	4	2	8	75	0
Cluster 2	5	3	10	80	0
Cluster 3	3	1	6	66.67	0
Cluster 4	2	1	4	75	0
Cluster 5	1	1	2	100	100
Cluster 6	1	1	2	100	100
Cluster 7	1	1	2	100	100
Cluster 8	4	1	8	62.5	0
Cluster 9	3	2	6	83.33	0
				742.50	300

จากตารางที่ 14 สามารถหาค่าต่าง ๆ ได้ดังต่อไปนี้

- คำนวณหาค่า ch(สูตรในบทที่ 2 หัวข้อ 2.3) ยกตัวอย่าง Cluster 1 สามารถหาได้จาก (4+2)/8 = 75 %
- คำนวณหาค่า igได้จาก Cluster ที่มีชื่อเดียวจะให้ค่า igเท่ากับ 100 ดังตัวอย่างใน Cluster 5, 6 และ 7
- ประสิทธิภาพของเนมแมทชิ่งอัลกอริทึมแบบ Included single cluster คือ742.50/9= 82.5%
- ประสิทธิภาพของเนมแมทชิ่งอัลกอริทึมแบบ Ignored single cluster คือ(742.50-300)/(9-3) = 73.75%

3.7 ขั้นตอนการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม

การเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมเป็นส่วนหนึ่งที่สามารถบอกได้ว่าอัลกอริทึมแต่ละอัลกอริทึมสามารถนำไปประยุกต์ใช้งานประเภทใด ดังนั้นผู้วิจัยจึงทำการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมทั้งหมด12 อัลกอริทึม ได้แก่ Soundex, Soundex2, Metaphone, Phonex, NYSIIS, Simplex, Guth, Levenshtein,I SG, LIG1, LIG2 และ LIG3 ด้วยวิธีการนำรายชื่อของ R&W บางส่วนมาเปรียบเทียบกัน โดยเนมแมทชิ่งอัลกอริทึมทั้ง 12 อัลกอริทึม ใช้รายชื่อเดียวกัน

3.8 ขั้นตอนการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

การเปรียบเทียบเวลาของวิธีการการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมทั้ง 3 วิธีการ ได้แก่ การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell(1998)วิธีการ F-Measure และเทคนิค Clustering เพื่อให้ทราบว่าวิธีการวัดประสิทธิภาพวิธีไหนใช้เวลาน้อยที่สุด ดังนั้น ผู้วิจัยจึงนำเนมแมทชิ่งอัลกอริทึมที่ใช้เวลาน้อยที่สุดและมีประสิทธิภาพสูงมาเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

บทที่ 4

ผลการวิจัย

ในบทนี้ จะแสดงผลการวิจัยที่ทำการทดลองตามขั้นตอนต่าง ๆ ในบทที่ 3 โดย ผลการวิจัยจะแบ่งเป็นหัวข้อต่าง ๆ ดังนี้

- ผลการวิเคราะห์เนมแมทชิ่งอัลกอริทึมแบบที่มีการเปรียบเทียบกับค่าThreshold
- ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Lait และ Randell
- ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-Measure
- ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยเทคนิค Clustering
- ผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม
- ผลการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

1. **ผลการวิเคราะห์เนมแมทชิ่งอัลกอริทึมในกลุ่มที่มีการเปรียบเทียบกับค่าThreshold**

จากบทที่ 3 ข้อ 3.5ได้มีการทดสอบหาค่า Threshold ที่เหมาะสมในการนำไปใช้กับการเปรียบเทียบชื่อด้วยเนมแมทชิ่งอัลกอริทึมทั้ง 6 อัลกอริทึม ได้แก่

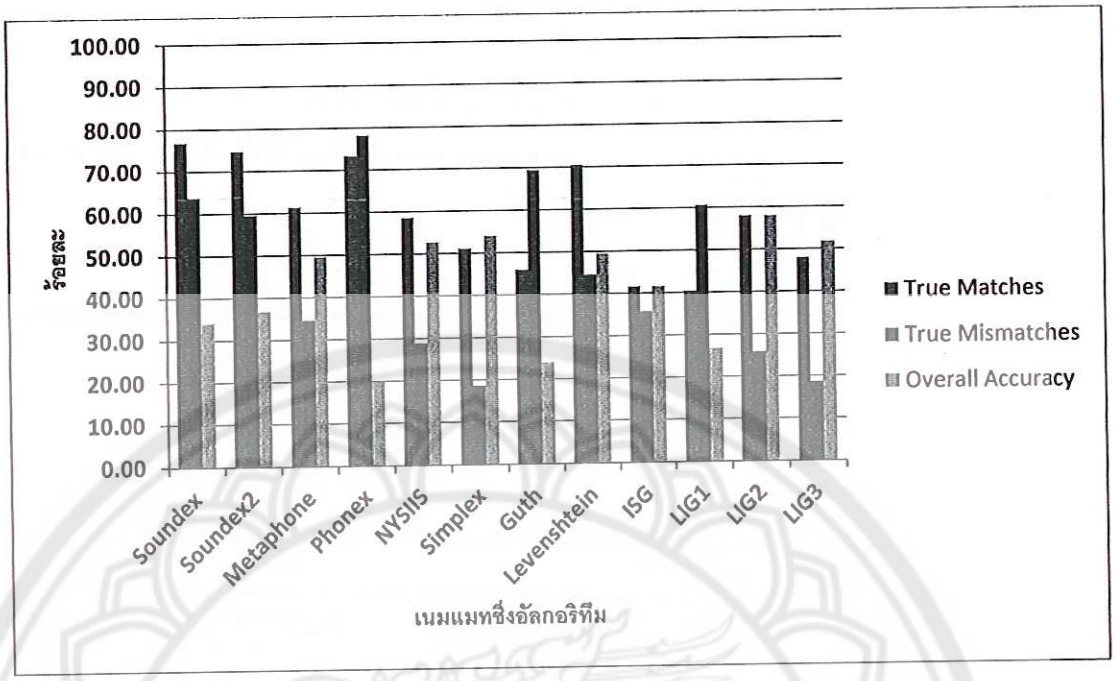
- Guth โดยค่า Threshold ที่เหมาะสมของ Guthคือ 80 % ของจำนวนตัวอักษรเฉลี่ยของ 2 ชื่อ
- Levenshtein โดยค่า Threshold ที่เหมาะสมของ Levenshteinคือ 30 % ของจำนวนตัวอักษรเฉลี่ยของ 2 ชื่อ
- ISG, LIG1, LIG2 และ LIG3 โดยค่า Threshold ที่เหมาะสมของ ISG, LIG1, LIG2 และ LIG3 คือ 0.8, 0.7, 0.8 และ 0.9 ตามลำดับ

2. **ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Lait และ Randell**

ผลการทดสอบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell (1998) ดังแสดงในตารางที่ 15 และ รูปที่ 3

ตารางที่ 15 แสดงผลการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell

	True Matches(%)	True Mismatches(%)	Overall Accuracy(%)
Soundex	76.84	63.69	34.06
Soundex2	74.68	59.51	36.81
Metaphone	61.32	46.40	39.96
Phonex	73.34	78.23	19.79
NYSIIS	58.55	28.90	52.68
Simplex	51.09	18.74	54.02
Guth	46.00	69.34	23.91
Levenshtein	70.50	44.52	49.38
ISG	41.57	35.70	41.53
LIG1	40.34	60.59	26.77
LIG2	58.00	25.92	57.98
LIG3	48.00	18.75	51.77



ภาพ 3 ผลการเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell

จากตารางที่ 15 และภาพ 3 การเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell (1998) พบว่าอัลกอริทึมที่มีค่า True Matches สูง ได้แก่ Soundex(76.84%), Soundex2 (74.68%) และ Phonex(73.34%) ตามลำดับ ส่วนอัลกอริทึมที่มีค่า True Mismatches ต่ำ ได้แก่ Simplex(18.74%), LIG3(18.75%), LIG2(25.92%) ตามลำดับ และอัลกอริทึมที่มีค่า Overall Accuracy สูงได้แก่ LIG2(57.98%), Simplex(54.02%) และ NYSIIS(52.68%) ตามลำดับ

การพิจารณาประสิทธิภาพของเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell(1998)สามารถพิจารณาได้จากค่า OverallAccuracy ซึ่งเนมแมทชิ่งอัลกอริทึมที่มีค่า Overall Accuracy สูง ได้แก่ LIG2 (57.98%), Simplex (54.02%) และ NYSIIS (52.68%) ตามลำดับ การที่มีค่า Overall Accuracy สูง มาจากการมีค่า True Matches สูง แต่ค่า True Mismatches ต่ำ ซึ่ง True Matches สูงแสดงว่าชื่อที่ Match กันตามวิธีการของเนมแมทชิ่งอัลกอริทึมนั้นมีความถูกต้องมาก และ True Mismatches ต่ำ แสดงว่าเนมแมทชิ่งอัลกอริทึมนั้นมีการ Match ที่ไม่ถูกต้องน้อย

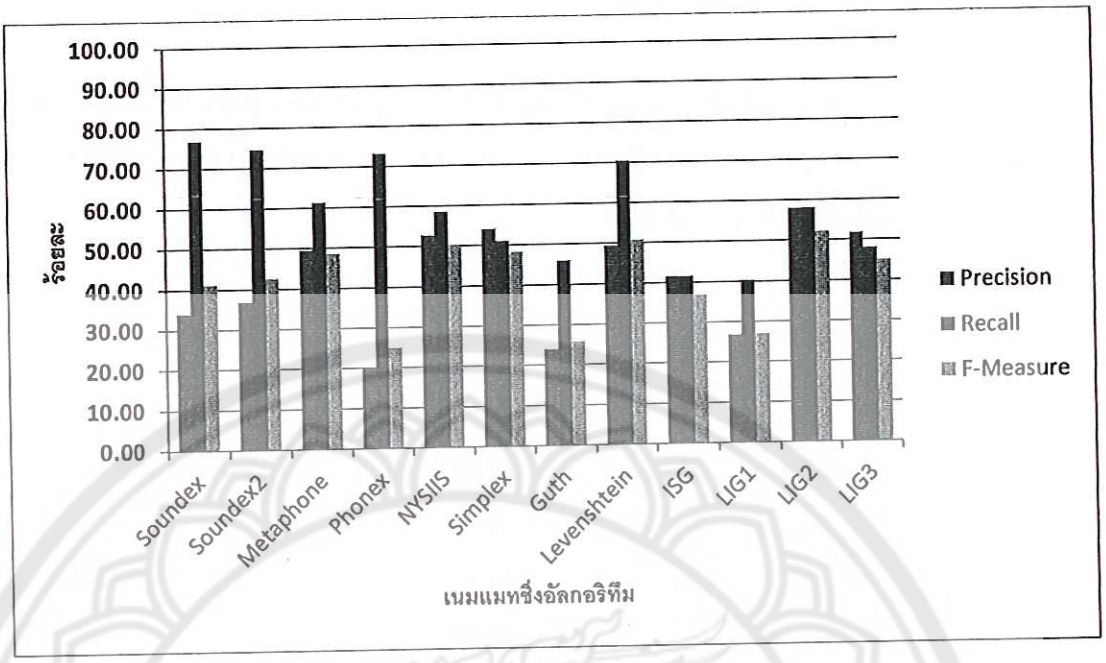
ส่วนเนมแมทชิ่งอัลกอริทึมที่มีค่า Overall Accuracy ต่ำ ได้แก่ Phonex(19.79%), Guth(23.91%)และ LIG1(26.77%)ตามลำดับ เนื่องจากเนมแมทชิ่งอัลกอริทึมทั้ง 3 อัลกอริทึมมีค่า True Mismatches สูงกว่าค่า True Matches ซึ่งแสดงให้เห็นว่าชื่อที่ Match มาทั้งหมดส่วนใหญ่ไม่ถูกต้อง จึงทำให้ค่า Overall Accuracy ต่ำ

3. ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-Measure

ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure ดังแสดงในตารางที่ 16 และภาพ 4

ตารางที่ 16 แสดงผลการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure

	Precision (%)	Recall (%)	F-Score (%)
Soundex	34.06	76.84	41.23
Soundex2	36.81	74.68	42.59
Metaphone	49.50	61.32	48.55
Phonex	19.79	73.34	24.97
NYSIIS	52.68	58.55	50.16
Simplex	54.02	51.09	48.30
Guth	23.91	46.00	25.80
Levenshtein	49.38	70.50	50.81
ISG	41.53	41.57	36.84
LIG1	26.77	40.34	26.89
LIG2	57.98	58.00	52.18
LIG3	51.77	48.00	45.01



ภาพ 4 ผลการเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure

จากตารางที่ 16และ ภาพ 4 การเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการF-measure พบว่าอัลกอริทึมที่ให้ค่า Precision สูง ได้แก่ LIG2(57.98%), Simplex(54.02%) และ NYSIIS(52.68%) ตามลำดับส่วนอัลกอริทึมที่ให้ค่า Recall สูง ได้แก่ Soundex(76.84%), Soundex2 (74.68%) และ Phonex(73.34%)ตามลำดับ และอัลกอริทึมที่ให้ค่า F-Score สูง ได้แก่ LIG2(52.18%), Levenshtein(50.81%) และ NYSIIS(50.16%) ตามลำดับ

การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-Measure นี้ สามารถพิจารณาได้จากค่า F-Score ซึ่งถ้าค่านี้สูงแสดงถึงประสิทธิภาพของเนมแมทชิ่งอัลกอริทึมสูงด้วย ซึ่งอัลกอริทึมที่มีค่า F-Score สูงได้แก่ LIG2 (52.18%), Levenshtein (50.81%) และ NYSIIS (50.16%) ตามลำดับ ซึ่งการที่มีค่า F-Score ที่สูงนั้น ต้องมีค่า Precisionสูงที่แสดงถึงความสามารถในการ Match ชื่อได้ถูกต้องและค่าRecallสูงที่แสดงถึงความสามารถในการค้นหาชื่อที่ Match

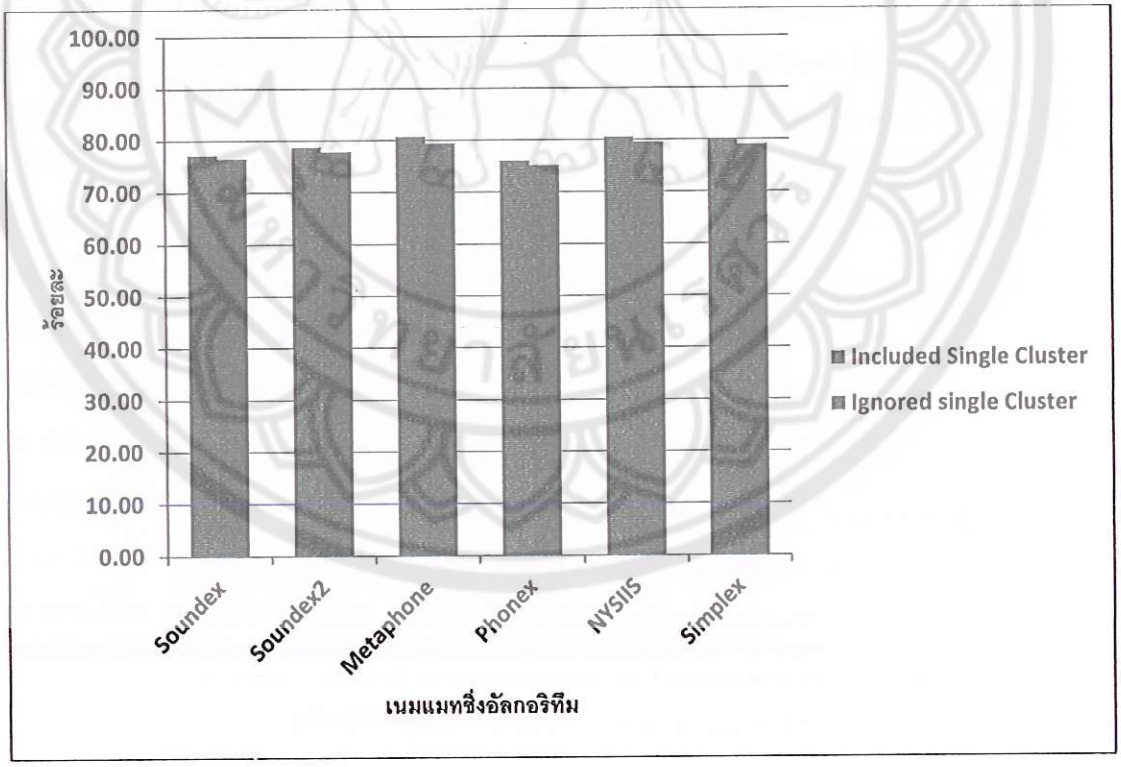
ส่วนเนมแมทชิ่งอัลกอริทึมที่มีค่า F-Score ต่ำ ได้แก่Phonex(24.97%), Guth(25.80%), LIG1(26.89%) ตามลำดับ ซึ่งอัลกอริทึมที่มีค่า F-Score ต่ำนั้นเกิดจากที่มีการ Match ชื่อไม่ถูกต้อง(False Positive) จำนวนมาก ทำให้ค่า Precision ต่ำ และ ยังไม่สามารถ Match กับชื่อที่ควรMatch ได้ (False Negative) ทำให้ค่า Recall ต่ำอีกด้วย

4. ผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering

ผลการทดสอบการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering ดังแสดงในตารางที่ 17 และภาพ 5

ตารางที่ 17 แสดงผลการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering

	Included Single Cluster	Ignored single Cluster
Soundex	77.26	76.60
Soundex2	78.73	77.85
Metaphone	80.71	79.40
Phonex	76.03	75.17
NYSIIS	80.51	79.52
Simplex	80.06	79.06



ภาพ5 ผลการเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิค Clustering

การตารางที่ 17 และภาพ 5 การเปรียบเทียบการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering พบว่าอัลกอริทึมที่ให้ผลการวัดประสิทธิภาพสูงในแบบ Included single cluster คือMetaphone(80.71%), NYSIIS(80.51%) และ Simplex(80.06%) ตามลำดับ และแบบ Ignored single cluster คือ NYSIIS(79.52%), Metaphone(79.40%) และ Simplex(79.06%) ตามลำดับ

จะเห็นได้ว่าเนมแมทชิ่งอัลกอริทึมที่ให้ผลการวัดประสิทธิภาพในแบบ Included single clusterที่สูงสุด กับเนมแมทชิ่งอัลกอริทึมที่ให้ผลการวัดประสิทธิภาพในแบบ Ignored single clusterที่สูงสุดไม่ใช่อัลกอริทึมเดียวกัน เนื่องจาก NYSIIS มีจำนวน single cluster มากกว่า Metaphone

สำหรับเนมแมทชิ่งอัลกอริทึมที่ให้ผลการวัดประสิทธิภาพทั้งในแบบ Included single cluster และ Ignored single clusterสูงใน 3 ลำดับแรกนั้น มาจากการเข้ารหัสที่เป็นตัวอักษรทั้งหมด ทำให้การเรียงลำดับตัวอักษรของรหัสเสียงสามารถจัดกลุ่มและสร้างCluster ได้ถูกต้องและมีประสิทธิภาพ

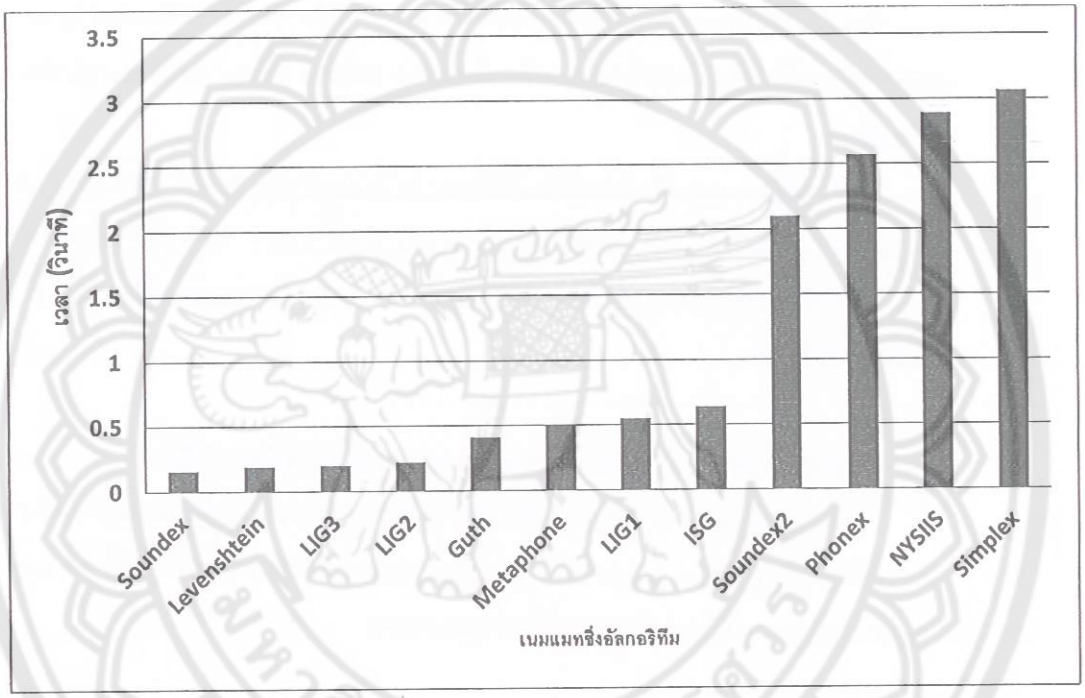
5. ผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม

จากบทที่ 3 ข้อ 3.8ผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมโดยการเปรียบเทียบ Base name กับ Variation nameบางส่วนในไฟล์รายชื่อ A Dictionary of English Surnamesดังแสดงในตารางที่ 18 และภาพ 6

ตารางที่ 18 แสดงเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม

เนมแมทชิ่งอัลกอริทึม	เวลา (วินาที)
Soundex	0.1615
Levenshtein	0.1930
LIG3	0.2001
LIG2	0.2222
Guth	0.4098
Metaphone	0.5061
LIG1	0.5505

เนมแมทชิ่งอัลกอริทึม	เวลา (วินาที)
ISG	0.6383
Soundex2	2.1029
Phonex	2.5698
NYSIIS	2.8894
Simplex	3.0659



ภาพ 6 ผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม

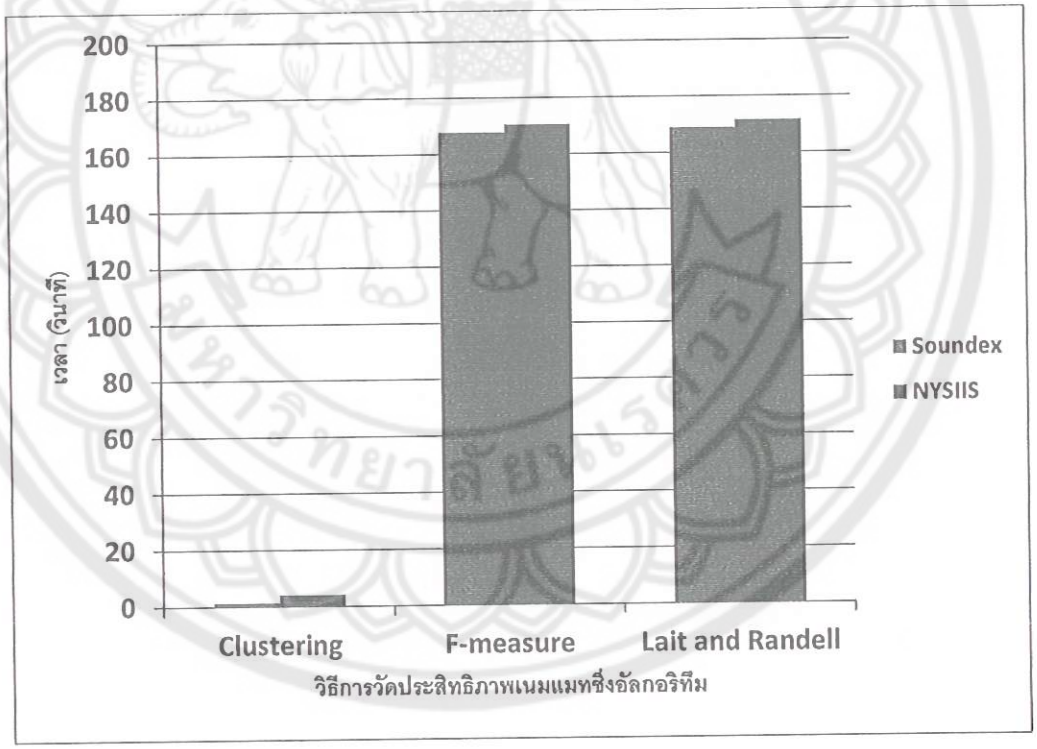
จากตารางที่ 18 และภาพ 6 การเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมพบว่าอัลกอริทึมที่มีเวลาในการประมวลผลน้อยที่สุด4 ลำดับ ดังนี้ Soundex, Levenshtein, LIG3 และ LIG2 ตามลำดับและเนมแมทชิ่งอัลกอริทึมที่มีเวลาในการประมวลผลมากที่สุด 4 ลำดับ ดังนี้ Simplex, NYSIIS, Phonexและ Soundex2 ตามลำดับ จะเห็นได้ว่า เนมแมทชิ่งอัลกอริทึมที่มีเวลาในการประมวลผลมากที่สุดจะอยู่ในกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียงมีเพียง Soundexและ Metaphoneที่มีเวลาในการประมวลผลน้อย เพราะ Soundexและ Metaphoneมีฟังก์ชันในการทำงานในภาษา PHP อยู่แล้ว จึงสามารถประมวลผลได้เร็วมากขึ้น

6. ผลการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

จากบทที่ 3 ข้อ 3.9 ผลการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมดังแสดงในตารางที่ 19 และภาพ 7

ตารางที่ 19 แสดงเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

วิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม	เวลา(วินาที)	
	Soundex	NYSIIS
Clustering	1.4843	4.1519
F-measure	167.8398	170.4523
Lait and Randell	168.9298	171.5695



ภาพ 7 ผลการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

จากตารางที่ 19 และ ภาพ 7 จะเห็นได้ว่ามีการนำเนมแมทชิ่งอัลกอริทึมที่มีเวลาในการประมวลผลน้อยที่สุดและมีประสิทธิภาพสูง ได้แก่ Soundex, NYSIISตามลำดับ มาทำการ

เปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมทั้ง 3 วิธีการ(Lait and Randell, F-Measure, Clustering)ซึ่งผลการทดลองพบว่า เวลาของวิธีการวัดประสิทธิภาพของ Soundex และ NYSIIS เป็นไปในทิศทางเดียวกัน คือ การวัดประสิทธิภาพด้วยเทคนิค Clustering มีเวลาน้อยที่สุด เนื่องจากมีการวนรอบคำนวณเพื่อสร้าง Cluster และหาค่าต่าง ๆ เพียงรอบเดียว ซึ่งแตกต่างจากการวัดประสิทธิภาพตามแนวคิดของ Laitและ RandellและวิธีการF-measure ที่ต้องวนรอบคำนวณหาค่าต่าง ๆ หลายรอบ

เหตุผลที่ไม่นำ LIG2 ที่มีประสิทธิภาพสูงสุดมาเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพ เนื่องจาก LIG2 ไม่สามารถวัดประสิทธิภาพด้วยเทคนิค Clustering ได้จึงเลือกใช้ NYSIIS ที่มีประสิทธิภาพรองลงมา

บทที่ 5

บทสรุป

การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมเป็นการนำเนมแมทชิ่งอัลกอริทึมทั้งหมด 12 อัลกอริทึม ได้แก่ Soundex, Soundex2, Phonex, Metaphone, Simplex, NYSIIS,Guth, Levenshtein,ISG, LIG1, LIG2 และ LIG3มาวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมทั้งหมด 3 วิธีการ ได้แก่ การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randell วิธีการ F-Measure และเทคนิค Clustering โดยใช้รายชื่อทั้งหมดของA Dictionary of English Surnames (Reaney and Wilson, 1997) ซึ่งเป็นพจนานุกรมนามสกุลภาษาอังกฤษที่มีความถูกต้องและน่าเชื่อถือ(Ground truth)พร้อมทั้งเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมและเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมซึ่งทั้งหมดได้พัฒนาด้วยภาษา PHP เนื่องจากภาษา PHP มีฟังก์ชันในการทำงานกับสายอักขระ (String) ได้ดี และยังมีฟังก์ชันที่เกี่ยวกับเนมแมทชิ่งอัลกอริทึมบางตัว ได้แก่ Soundex, Metaphoneและ Levenshtein

1. สรุปผลการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึม

เนมแมทชิ่งอัลกอริทึมที่มีประสิทธิภาพสูงตามแนวคิดของ Laitและ Randell (1998)ให้พิจารณาได้จากค่า Overall Accuracy ซึ่งอัลกอริทึมที่ให้ค่า Overall Accuracy สูงสุดคือ LIG2 เนื่องจาก มีค่า True Matches สูง ซึ่งแสดงว่ามีความถูกต้องของการ Match มากและ มีค่าTrue Mismatches ต่ำ ซึ่งแสดงว่ามีการ Match ที่ไม่ถูกต้องน้อย

การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure นี้ สามารถพิจารณาได้จากค่า F-score ซึ่งอัลกอริทึมที่มีค่าF-Score สูงสุด คือ LIG2 เนื่องจากมีค่า Precision กับ ค่า Recall สูงและใกล้เคียงกัน ซึ่งแสดงให้เห็นว่า LIG2 มีทั้งความสามารถในการ Matchชื่อได้ถูกต้องและ ความสามารถในการค้นหาชื่อที่ Match

การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยเทคนิคClustering มีด้วยกัน2 แบบ คือ Included single clusterและ Ignored single clusterโดยแบบIgnored single clusterจะให้ผลที่มีความน่าเชื่อถือมากกว่าแบบ Included single clusterเนื่องจาก มีการตัด Single cluster หรือ cluster ที่มีชื่อเดียวออก เพราะ Single cluster จะให้ค่า ch ของ Cluster นั้นเป็น 100% และถ้ามีจำนวนมากจะทำให้ผลของการวัด

ประสิทธิภาพมีค่ามากขึ้น ซึ่งอาจทำให้การวัดประสิทธิภาพมีความคลาดเคลื่อนได้และเนมแมทชิ่งอัลกอริทึมที่มีประสิทธิภาพในแบบ Ignored single cluster สูงสุด คือ NYSIIS

2. สรุปภาพรวมของการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึม

จากผลการวัดประสิทธิภาพและเปรียบเทียบทั้งหมด 3 วิธีการ มีเพียงเนมแมทชิ่งอัลกอริทึมในกลุ่มที่มีการเปรียบเทียบกับค่าThreshold ทั้งหมด 6 อัลกอริทึม ได้แก่ Soundex, Soundex2, Metaphone, Phonex, NYSIIS และ Simplex ที่สามารถทำการวัดประสิทธิภาพได้ครบทั้ง 3 วิธีการ โดยอัลกอริทึมที่มีประสิทธิภาพสูง ได้แก่ NYSIIS และ Simplex ซึ่งผลการทดลองสอดคล้องกันทั้ง3 วิธีการ

อย่างไรก็ตาม การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมที่สามารถวัดประสิทธิภาพได้ทั้งหมด 12 อัลกอริทึม มีเพียง 2 วิธีการ ได้แก่ การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมตามแนวคิดของ Laitและ Randellและวิธีการ F-Measure โดยผลการวัดประสิทธิภาพพบว่า LIG2 มีประสิทธิภาพสูงที่สุด ทั้ง 2 วิธีการ

การวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมทั้ง 12 อัลกอริทึมโดยใช้การวัดประสิทธิภาพทั้งหมด 3 วิธีการ สรุปได้ว่าเนมแมทชิ่งอัลกอริทึมแต่ละอัลกอริทึมมีจุดเด่นในแต่ละเรื่องไม่เหมือนกัน เพราะฉะนั้นการเลือกใช้เนมแมทชิ่งอัลกอริทึมไปประยุกต์ใช้กับงานแต่ละงานควรเลือกให้เหมาะสม เช่น Search Engine ควรใช้อัลกอริทึมที่สามารถค้นหาชื่อออกมาให้ได้มากที่สุดซึ่งอาจพิจารณาจากค่า True Matches ดังนั้น เนมแมทชิ่งอัลกอริทึมที่เหมาะสม คือ Soundex, Soundex2, Phonexและ Levenshteinหรือการค้นหาชื่อในงานทะเบียนต่าง ๆ อาจต้องการความถูกต้องในการค้นหา ซึ่งพิจารณาได้จากค่า Overall Accuracy และค่า F-Scoreโดยเนมแมทชิ่งอัลกอริทึมที่เหมาะสมคือ LIG2และ NYSIIS

3. สรุปผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึม

จากผลการทดลอง สรุปได้ว่าเนมแมทชิ่งอัลกอริทึมที่มีเวลาในการประมวลผลน้อยจะอยู่ในกลุ่มที่การเปรียบเทียบกับค่าThreshold ซึ่งมีเพียง Soundexและ Metaphoneเท่านั้นที่อยู่ในกลุ่มของการเปลี่ยนชื่อเป็นรหัสเสียง เนื่องจาก มีการใช้ฟังก์ชันของภาษา PHP จึงทำให้มีเวลาในการประมวลผลน้อยลง

4. สรุปผลการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึม

การเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมได้นำอัลกอริทึมที่มีเวลาในการประมวลผลน้อยที่สุดและมีประสิทธิภาพสูงมาใช้ในการเปรียบเทียบเวลาของวิธีการวัดประสิทธิภาพ

ซึ่งจากผลการทดลอง สรุปได้ว่า การวัดประสิทธิภาพที่มีเวลาน้อยที่สุด คือ การวัดประสิทธิภาพด้วยเทคนิค Clustering ดังนั้น ในการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมควรนำเทคนิค Clustering ไปใช้

5. ข้อเสนอแนะ

จากการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมทั้ง 12 อัลกอริทึมจะเห็นได้ว่าผลการทดสอบที่ได้มีความน่าเชื่อถือ และตรงตามขอบเขตที่ได้กำหนดไว้ แต่อย่างไรก็ตามข้อเสนอแนะที่ควรมีการปรับปรุงให้ดียิ่งขึ้น มีดังต่อไปนี้

1) ควรมีการวัดประสิทธิภาพในภาษาอื่น ๆ นอกเหนือจากภาษาอังกฤษเพิ่มมากขึ้นเช่น ภาษาไทย
2) ควรมีการนำภาษาคอมพิวเตอร์อื่นมาใช้ในการทดลองเพื่อผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมมีความน่าเชื่อถือมากยิ่งขึ้น เช่น ภาษา C

จากข้อเสนอแนะที่กล่าวมาข้างต้น สรุปได้ว่า การวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทย อาจจะใช้รายชื่อที่มีความถูกต้องและน่าเชื่อถือ เช่น รายชื่อจากทะเบียนทะเบียนราษฎร์รายชื่อจาก Family Tree ของคนไทย เป็นต้นมาวัดประสิทธิภาพด้วยวิธีการที่มีความแตกต่างกันได้ ซึ่งการประยุกต์ใช้เนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทยนั้น Snaeและ Bruckner (2009) ได้พัฒนาเนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทย คือ Metasoundซึ่งนำไปใช้กับการตั้งชื่อและเปลี่ยนชื่อไทยให้สอดคล้องกับชื่อบิดาและมารดา หรือเปลี่ยนชื่อให้มีความคล้ายคลึงกับชื่อเดิม นอกจากนี้ Panawongและ Snae (2010) ได้พัฒนาระบบค้นหาสถานที่ท่องเที่ยวในประเทศไทยโดยได้มีการนำเนมแมทชิ่งอัลกอริทึมมาช่วยในการสืบค้นสถานที่ท่องเที่ยวที่มีการพิมพ์หรือสะกดที่คล้ายคลึงกัน เพื่อให้ได้ข้อมูลครอบคลุมมากยิ่งขึ้น

เอกสารอ้างอิงของโครงการวิจัย

ไพศาล โมลิสกุลมงคล. (2544). พัฒนา Web Database ด้วย PHP.กรุงเทพฯ: ไทยเจริญการพิมพ์.

Bouchard, G. and Pouyez, C. (1980). Name Variations and Computerised Record Linkage. Historical Methods, 13 (2), 119-125.

Christen, P. (2006). A Comparison of Personal Name Matching: Techniques and Practical Issues. Australian National University.

Davey, C. and Javis, A.S. (1990). Microcomputers for Microhistory: a database approach to the construction of small English populations. History and Computing, 2(3), 187-193.

Gill, L.E., Goldacre, M.J., Simmons, H.M., Bettley, G.A. and Griffith, M. (1993).Computerised Linkage of Medical Records: Methodological Guidelines.Journal of Epidemiology and Community Health, 47, 316-319.

Gill, L.E. OX-LINK (1997).The Oxford Medical Record Linkage System, Complex linkage made easy, Record Linkage Techniques – 1997: Proceedings of an International Workshop and Exposition, VA, 15-33.

Guth, G.J.A (1976). Surname Spellings and Computerized Record Linkage. Historical Methods Newsletter, 10 (1), 10-19.

Halkidi, M., Batistakis, Y. and Vazirgiannis, M. (2001).Cluster algorithms and validity measures.Tutorial paper, proceedings of the SSDBM. Virginia, USA.

Lait, A. J. and Randell, B. 1998. An Assessment of Name Matching Algorithm. Society of Indexers Genealogical Group, Newsletter Contents. SIGGNL issues 17. 1-31.

Levenshtein, V.I. (1965). Binary codes capable of correcting deletions.Insertions and reversals.DokladyAkademiiNauk SSSR, 163(4), 845-848.

Lynch, B.T. and Arends, W.L. (1977). Selection of a Surname Encoding Procedure for the Statistical Reporting Service Record Linkage System. Washington, DC: United States Department of Agriculture.

Panawong, N. andSnae, C., Search System for Attractions in Thailand with Ontology and Name Matching, Journal of Information Science and Technology, Vol.1, No.2 , pp. 60-69,2010

Newcombe, H.B., Kennedy, J.M., Axford, S.J. and James, A.P (1990).Automatic Linking of Vital Records.Science, 130, 954-959.

Ramkumar, G.D. and Swami, A.N. (1998). Clustering Data Without Distance Functions. Data Engineering Bulletin, 21(1), 9-14.

Reaney, P.H. and Wilson, R.M. (1997).A Dictionary of English Surnames.Oxford University Press.

Snae, C. (2006). An Investigation of Computer based Nominal Data Record Linkage.Ph.D. Dissertation, University of Liverpool.

Snae, C. (2007). A Comparison and Analysis of Name Matching Algorithm.World Academy of Science, 25, 252-257.

Snae, C. and Brueckner, M. (2006). Concept and Rule Based Naming System. The Journal of Issues in Informing Science and Information Technology, 3, 619-634.

Snae, C. and Brueckner, M., "LOWCOST: Local organization search with consolidated ontologies for name, space and time", International Conference on Software Engineering, Innsbruck, Austria, Feb. 2007, pp.13-15

Snae, C. and Diaz, B.M. (2002).An Interface for Mining Genealogical Nominal Data Using the Concept of linkage and a Hybrid Name Matching Algorithm.Journal of 3D-Forum Society, 16(1), 142-147.

Snae, C., Singhadech, N., Emapana, B., and Brueckner, M. (2006).Interactives Transliteration Tools for Explanation Level Language System (IT-TELLS), Proceedings of the ITC-CSCC 2006: The 21st International Technical Conference on Circuits/Systems, Computers and Communications, Chiang Mai, Thailand, 10-13 July 2006.

Thaller, M. (1987).Method and Techniques of Historical Computation, in History and Computing, ed. P. Denley and D. Hopkin, Manchester Uni. Press, 147-156.

Winchester, I. (1970). The Linkage of Historical Records by Man and Computer: Techniques and Problems. Journal of Interdisciplinary History, 1, 107-124.

Output ที่ได้จากโครงการ

จักรกฤษณ์ เสน่ห์ นมะหุต . (กรกฎคม-ธันวาคม/2558) . ชื่อบทความ. วารสารบริหารธุรกิจเทคโนโลยีมหานคร. กรกฎคม-ธันวาคม/2558 (2) : 1-22. (Impact Factor 0.1)

ภาคผนวก

มหาวิทยาลัยเทคโนโลยีมหานคร
MAHANAKORN UNIVERSITY OF TECHNOLOGY

ที่ มทม (บธ) ๕๗ / ๐๖๐

๑๕ กันยายน ๒๕๕๗

เรื่อง แจ้งการตีพิมพ์บทความ

เรียน ผู้ช่วยศาสตราจารย์ ดร.จักรกฤษณ์ เสน่ห์ นมะหุต

ตามที่ กองบรรณาธิการวารสารบริหารธุรกิจเทคโนโลยีมหานคร ได้รับบทความจากท่านเป็นที่เรียบร้อยแล้ว นั้น กองบรรณาธิการฯ จึงได้ส่งบทความฉบับสมบูรณ์ของท่าน ชื่อ "**การวิเคราะห์ประสิทธิภาพของ Name Matching Algorithm สำหรับระบบสืบค้นข้อมูลท่องเที่ยว**" เข้าสู่กระบวนการประเมินบทความโดยคณะผู้ทรงคุณวุฒิ และท่านได้ดำเนินการปรับแก้ตามผลการประเมินเป็นที่เรียบร้อยแล้ว ดังนั้นคณะผู้ทรงคุณวุฒิดังกล่าว จึงมีมติให้กองบรรณาธิการฯ ดำเนินการตีพิมพ์บทความดังกล่าวได้

กองบรรณาธิการฯ จึงพร้อมตีพิมพ์บทความของท่านลงในวารสารบริหารธุรกิจเทคโนโลยีมหานคร ภายใน ปีที่ ๑๒ ฉบับที่ ๒ (กรกฎาคม – ธันวาคม ๒๕๕๘)

จึงเรียนมาเพื่อทราบ

ขอแสดงความนับถือ

ชิตวันพัทน์ วีระสัย

(อาจารย์ ชิตวันพัทณ์ วีระสัย)
บรรณาธิการวารสารบริหารธุรกิจเทคโนโลยีมหานคร

140 ถนนเชื่อมสัมพันธ์ แขวงการะหุ่งราย เขตหนองจอก กรุงเทพฯ 10530 โทร.0-2988-3655, 0-2988-3666 โทรสาร 0-2988-4040
140 CHEUM-SAMPAN RD., NONG CHOK, BANGKOK 10530 THAILAND Tel. +66(0)-2988-3655, +66(0)-2988-3666 Fax. +66(0)-2988-4040 www.mut.ac.th

วารสารบริหารธุรกิจเทคโนโลยีมหานคร
MUT Journal of Business Administration
ปีที่ 12 ฉบับที่ 2 (กรกฎาคม 2558 – ธันวาคม 2558)
Volume 12 Number 2 (July 2015 – December 2015)

การวิเคราะห์ประสิทธิภาพของอัลกอริทึม Name Matching สำหรับระบบสืบค้นข้อมูลท่องเที่ยว

Performance Analysis of Name Matching Algorithm for Tourist Information Searching System

ผศ.ดร.จักรกฤษณ์ เสน่ห์ นมะหุต[1,*]

[1] Ph.D., ผู้ช่วยศาสตราจารย์ ภาควิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์ มหาวิทยาลัยนเรศวร อ.เมือง จ.พิษณุโลก 65000

บทคัดย่อ

เมื่อนักท่องเที่ยวต้องการสืบค้นข้อมูลท่องเที่ยวส่วนมากจะทำการสืบค้นผ่านเสิร์ชเอนจินที่ได้รับความนิยมอย่างกูเกิ้ล แต่ผลลัพธ์ที่ได้นั้นมีจำนวนมากอาจต้องเสียเวลาในการเข้าถึงข้อมูลจากหลายเว็บไซต์และผลลัพธ์ที่ได้อาจไม่ตรงกับความต้องการ อีกทั้งในบางเว็บไซต์นำเสนอข้อมูลบนเว็บไซต์ที่อาจมีการเขียนไม่ถูกต้องตามหลักภาษาอังกฤษก็จะทำให้ได้ผลของการสืบค้นไม่พบเว็บไซต์ที่เขียนไม่ถูกต้อง ดังนั้นผู้วิจัยจึงได้ประยุกต์ใช้อัลกอริทึมเนมแมชชิ่งมาใช้ในระบบสืบค้นข้อมูลท่องเที่ยวเพื่อแก้ปัญหาในการกรองข้อมูลชื่อที่สามารถเขียนได้หลายรูปแบบหรือมีความคล้ายคลึงกันทั้งคำพ้องรูป พ้องเสียงหรือสะกดผิดในภาษาอังกฤษ ทำให้ข้อมูลท่องเที่ยว เช่น ชื่อของสถานที่ หรือชื่อของจังหวัด ที่เขียนผิดถูกค้นพบ อีกทั้งยังได้ทำการวัดประสิทธิภาพของอัลกอริทึมเนมแมชชิ่ง โดยใช้ F-Measure ซึ่งจากผลการวิจัยพบว่า อัลกอริทึม Soudex มีประสิทธิภาพดีที่สุดในการสืบค้นข้อมูลท่องเที่ยวที่สามารถ ช่วยในการแก้ไขปัญหาการพิมพ์ผิดหรือไม่ถูกหลักภาษาอังกฤษซึ่งจะทำให้ไม่พลาดข้อมูลที่ต้องการ

* E-mail address: Chakkrits@nu.ac.th

คำสำคัญ: การสืบค้นข้อมูล ข้อมูลท่องเที่ยว ระบบสืบค้นข้อมูล อัลกอริทึมเนมแมทชิ่ง การวัดประสิทธิภาพ รหัสเสียง

ABSTRACT

To extract tourism information, most travelers rely on web search engines such as Google, so they are faced with a huge number of web pages as results. Consequently, they apply a time consuming process accessing many websites but might still not get what they need. Moreover, if keywords or Web site contents are misspelled in English or other languages, the search results will consist of incomplete or incorrect information. To deal with these problems, name matching algorithms are used to make connections between several similar forms in English words, such as homographs, homophones and misspellings. This helps discover mistakes of tourists' information such as names of locations and provinces. In order to measure the performance of the name matching algorithms, F-Measure is used. The results show that Soundex algorithm has the best performance in tourist information searching, which can solve misspelling of information with names in English language effectively and will not miss any tourist information.

Keywords: Information Retrieval, Tourist information, information searching sysem, Name Matching Algorithm, performance measurement, Sound Code

---

บทนำ

การท่องเที่ยวถือเป็นอุตสาหกรรมบริการที่ทำให้ประเทศมีรายได้เพิ่มขึ้นและยังมีบทบาทต่อการพัฒนาเศรษฐกิจของชาติอีกด้วย ทั้งนี้ในปัจจุบันพบว่ามีข้อมูลท่องเที่ยวเป็นจำนวนมากที่ถูกนำเสนอบนอินเทอร์เน็ตและมีรูปแบบการนำเสนอที่แตกต่างกันออกไป เช่น รูปแบบข้อความ รูปแบบมัลติมีเดีย เป็นต้น ซึ่งในการค้นหาข้อมูลท่องเที่ยวเพื่อให้ได้ตรงกับความต้องการของนักท่องเที่ยวนั้นจึงมีความสำคัญต่อนักท่องเที่ยวเป็นอย่างมาก โดยเฉพาะการป้อนคำค้นหาที่บางครั้งมีการเขียนผิดโดยไม่ตั้งใจหรือการเขียนหรือสะกดคำไม่ถูกต้องตามหลักภาษาไทยหรือภาษาอังกฤษ (Snae et al., 2007) หรือมีการใช้คำค้นหาที่เป็นชื่อเรียกอื่นๆ ของสถานที่นั้นๆ เช่น วัดใหญ่ หมายถึง วัดพระศรีรัตนมหาธาตุวรมหาวิหาร สองแคว หมายถึง จังหวัดพิษณุโลก เป็นต้น ซึ่งทำให้ได้ผลลัพธ์ตรงและไม่ต้องกับความต้องการปะปนกันอยู่

การค้นหาข้อมูลท่องเที่ยวอาจมาจากหลายๆ แหล่ง เช่น สื่อสิ่งพิมพ์ แผ่นพับ อินเทอร์เน็ต บางครั้งมีการใช้คำค้นหาที่เป็นชื่อเรียกอื่นๆ ที่ใช้เรียกกันมาตั้งแต่สมัยโบราณของสถานที่นั้นๆ แต่ชื่อเหล่านี้หมายถึงสถานที่เดียวกัน โดยมีชื่อเรียกที่หลากหลายตามวัฒนธรรม พื้นเพ ภูมิลำเนา ลักษณะภูมิประเทศ เช่น วัดใหญ่ หมายถึง วัดพระศรีรัตนมหาธาตุวรมหาวิหาร เมืองสองแคว เมืองอกแตก เมืองพุทธชินราช หมายถึง จังหวัดพิษณุโลก บางครั้งการค้นหาอาจจำเป็นต้องใช้ส่วนประกอบอื่นๆ เช่น คำขวัญ ดอกไม้ ต้นไม้ สัญลักษณ์ ประจำจังหวัดนั้นๆ ทำให้เกิดการค้นหาข้อมูลที่พลาดไป (นฤพนธ์และจักรกฤษณ์, 2554) หรือ บางครั้งในการค้นหาข้อมูลท่องเที่ยวเกี่ยวกับจังหวัดนั้นๆ สถานที่ท่องเที่ยว อาจมีใช้คำค้นหาที่มีการเขียนผิดหรือไม่ถูกต้องตามหลักภาษาไทยและภาษาอังกฤษซึ่งคำเหล่านั้นปรากฏอยู่มากมายในเว็บไซต์ท่องเที่ยว เช่น ค้นหาข้อมูลท่องเที่ยวในจังหวัดพิษณุโลก (Phitsanulok) แต่พบว่าบางเว็บไซต์ใช้คำว่า "พิจสะนุโลก" หรือ "พิษนุโลก" หรือ "พิดโลก" หรือ "พิดสะณุโลก" ซึ่งเป็นการเขียนชื่อจังหวัดพิษณุโลกไม่ถูกต้องตามหลักภาษาไทย และพบว่าบางเว็บไซต์ใช้คำว่า "Phisanulok" หรือ "pitsanulok" หรือ "pisanulok" หรือ "phitsanurok" ซึ่งเป็นการเขียนชื่อจังหวัดพิษณุโลกไม่ถูกต้องตามหลักภาษาอังกฤษ (Snae and Brueckner, 2007) ซึ่งชื่อที่มีการสะกดผิดหรือเขียนผิดนั้นก็ย่อมสามารถก่อให้เกิดปัญหาในการค้นหาสถานที่ท่องเที่ยวได้ ซึ่งถ้าการบันทึกชื่อที่สะกดผิดหรือการเขียนที่แตกต่างกัน แม้เพียงเล็กน้อยจะสามารถนำไปสู่ความผิดพลาดในการระบุสถานที่ท่องเที่ยวเนื่องจากชื่อที่เขียนแตกต่างกันสองชื่อนั้นอาจบ่งชี้ถึงสถานที่เดียวกันก็เป็นได้ โดย Snae and Brueckner (2006) ได้แบ่งความคลาดเคลื่อนที่อาจเกิดขึ้นจากการค้นหาชื่อที่สามารถออกเสียงและการสะกดที่แตกต่างกันได้ 3 ประเภท ดังต่อไปนี้

1) ความคลาดเคลื่อนจากรูปแบบตัวอักษร

ปัญหานี้เกิดจากการใช้ตัวพิมพ์ใหญ่, เครื่องหมายวรรคตอน, การเว้นวรรค, คำขยายความและการย่อ

2) ความคลาดเคลื่อนจากรูปแบบการสะกดคำ

ความคลาดเคลื่อนนี้จะขึ้นอยู่กับการสันนิษฐานที่มาและชื่อเป้าหมายที่มีความแตกต่างกันเนื่องจากความผิดพลาดในการคัดลอกเช่นการออกเสียงที่ต่างออกไปรวมถึงการสะกดคำผิดและการพิมพ์ผิดสิ่งคลาดเคลื่อนเหล่านี้แม้จะไม่ได้ส่งผลถึงการออกเสียงแต่ก็ก่อให้เกิดปัญหาในการจับคู่ของชื่อได้

3) ความคลาดเคลื่อนจากรูปแบบการออกเสียง

ความคลาดเคลื่อนจากการออกเสียงเกิดจากการที่โครงสร้างของชื่อถูกเปลี่ยนแปลงไปด้วยการออกเสียงเช่นการสะกดคำว่า"Pooh" ในภาษาอังกฤษแต่ในภาษาเยอรมันจะสะกดเป็น"Phu" แทนซึ่งการออกเสียงที่เปลี่ยนแปลงไปนี้อาจเกิดจากการได้ยินผิดจึงทำให้โครงสร้างของชื่อมีการเปลี่ยนแปลงไปเช่นคำว่า"อยุธยา" และ"อโยธยา" เป็นชื่อที่เกี่ยวข้องกันแต่มีการออกเสียงแตกต่างกันเป็นอย่างมากซึ่งจริงๆแล้วรูปแบบการออกเสียงของชื่อนั้นสามารถแสดงได้หลายรูปแบบเช่นอาจแสดงเป็น "พิษณุโลก" หรืออีกแบบที่ได้ถูกย่อให้สั้นเป็น"พิดโลก" หรือ "ราชบุรี" เป็น "ราดรี" เป็นต้น

ในงานวิจัยนี้ได้ทำการวิเคราะห์ ประสิทธิภาพของอัลกอริทึม Name Matching ที่อยู่ในกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง ได้แก่ Soundex, Soundex2, Phonex, Metaphone, Simplex, และ NYSIIS มาทำการทดสอบกับฐานข้อมูลชื่อจังหวัดในประเทศไทยจากที่ได้จากการสืบค้นข้อมูลจากกูเกิ้ล เพื่อนำมาใช้ในการแก้ปัญหาชื่อที่มีการเขียนผิดหรือไม่ถูกต้องตามหลักภาษาอังกฤษ เช่น ความคล้ายคลึงกันของตัวสะกดที่สามารถเขียนแทนกันได้ และคำพ้องรูป พ้องเสียง ทำให้ข้อมูลท่องเที่ยวที่เขียนผิดสามารถถูกค้นพบได้ในกูเกิ้ล อีกทั้งยังสามารถนำเอาอัลกอริทึมเนมแมทชิ่งที่มีประสิทธิภาพดีที่สุดมาทำการแก้ปัญหากับข้อมูลท่องเที่ยวที่มีการเขียนผิดในภาษาอังกฤษที่มีอยู่มากมายในเว็บไซต์ท่องเที่ยวต่าง ๆ เพื่อที่จะสามารถเพิ่มประสิทธิภาพของ ระบบสืบค้นข้อมูลท่องเที่ยวได้อีกด้วย

วัตถุประสงค์ของการวิจัย

1. เพื่อพัฒนาและวิเคราะห์ประสิทธิภาพของอัลกอริทึมเนมแมชชิ่งต่างๆที่นำมาใช้ในระบบสืบค้นข้อมูลท่องเที่ยว เพื่อนำมาแก้ปัญหาคำที่มีความหลากหลายและเขียนผิดหรือไม่ถูกต้องตามหลักภาษาอังกฤษ ด้วยหลักการ F-measure
2. เพื่อประยุกต์ใช้อัลกอริทึมที่มีประสิทธิภาพที่ดีที่สุดที่ได้จากข้อที่ 1 มาทำการออกแบบและพัฒนาระบบสืบค้นข้อมูลท่องเที่ยวที่สามารถตอบสนองความต้องการในการค้นหาอย่างมีประสิทธิภาพ

ประโยชน์ของการวิจัย

1. ได้เนมแมชชิ่งอัลกอริทึมที่มีประสิทธิภาพดีที่สุดในการสืบค้นข้อมูลที่สามารถ ช่วยในการแก้ไขปัญหาการพิมพ์ผิดหรือไม่ถูกหลักภาษาอังกฤษซึ่งจะทำให้ไม่พลาดข้อมูลที่ต้องการและสามารถทำงานผ่านเว็บแอพลิเคชันได้
2. ได้ระบบสืบค้นข้อมูลท่องเที่ยวด้วยเนมแมชชิ่งที่มีประสิทธิภาพ ที่สามารถช่วยอำนวยความสะดวกต่อนักท่องเที่ยว อีกทั้งยังช่วยส่งเสริมและพัฒนาตลาดการท่องเที่ยวและการบริการให้เพิ่มมากขึ้น
3. ได้ระบบค้นหาข้อมูลท่องเที่ยวที่มีประสิทธิภาพที่สามารถช่วย
    - เพิ่มการส่งเสริมและพัฒนาตลาดการท่องเที่ยวและการบริการ
    - ลดค่าใช้จ่ายในการแสวงหาข้อมูลที่ถูกต้อง และครบถ้วน ของผู้ใช้บริการ
    - ช่วยกระตุ้นการท่องเที่ยวได้มากขึ้น อีกทั้งยังเป็นการเพิ่มยอดค่าบริการต่าง ๆ ด้วย

การทบทวนวรรณกรรม

การใช้อินเทอร์เน็ตเพื่อการสืบค้นข้อมูลได้รับความนิยมอย่างแพร่หลาย เนื่องจากถือได้ว่าเป็นคลังข้อมูลที่ใหญ่ โดยเฉพาะข้อมูลทางด้านการท่องเที่ยวที่มีความสำคัญต่อนักท่องเที่ยวเป็นอย่างมาก โดยมีงานวิจัยที่ได้นำเทคโนโลยีและแนวคิดต่าง ๆ มาช่วยนำเสนอข้อมูลด้านการท่องเที่ยวให้มีประสิทธิภาพ ซึ่งในปัจจุบันมีเว็บแนะนำการท่องเที่ยวและเว็บที่ให้ข้อมูลการท่องเที่ยวอย่างมากมาย เช่น Web Application สำหรับให้บริการข้อมูลแหล่งเที่ยวเที่ยว ที่พัก ร้านค้า แผนที่ และวางแผนการท่องเที่ยวแก่นักท่องเที่ยวหรือผู้ที่สนใจจะท่องเที่ยวยังสถานที่ต่าง ๆ ซึ่งสำหรับการบริการสืบค้นข้อมูลท่องเที่ยว แผนที่ของสถานที่ท่องเที่ยวถือว่าเป็นข้อมูลท่องเที่ยวที่มีความสำคัญมากในการนำไปใช้แสดงผลการสืบค้นของระบบการให้บริการข้อมูลท่องเที่ยว โดยส่วนใหญ่จะอยู่ในรูปแบบ ระบบสารสนเทศภูมิศาสตร์ (Geographic Information System: GIS) ซึ่งจะเห็นได้จาก งานวิจัยของ สุรางค์ รัตน์ และ จักรกฤษณ์ (2552) ได้นำเสนอระบบบริหารการจัดการข้อมูลและเชื่อมโยงเส้นทางแหล่งท่องเที่ยวในรูปแบบอนิเมชัน)Animation) เพื่อดึงดูดนักท่องเที่ยว ซึ่งใช้แผนที่ระบบสารสนเทศภูมิศาสตร์ (Geographic Information System: GIS) ในการเชื่อมโยงเส้นทางการเดินทางไปแหล่งท่องเที่ยวและสถานที่ให้บริการต่าง ๆ ภายในเขตภาคเหนือ ประกอบด้วย จังหวัดพิษณุโลก สุโขทัย เพชรบูรณ์ และอุตรดิตถ์ โดยใช้ Google Map ในการนำเสนอแผนที่เชื่อมโยงเส้นทางแหล่งท่องเที่ยว รวมทั้งเสนอแหล่งท่องเที่ยวที่ใกล้เคียงเพื่อดึงดูดความสนใจของนักท่องเที่ยว ผลการทำงานทำให้วางแผนการเดินทางได้หลายทางเลือกตามความต้องการ สะดวก รวดเร็ว และยังใช้เป็นต้นแบบสำหรับภูมิภาคอื่น ๆ ได้อีกด้วย ข้อดีของระบบนี้คือมีความยืดหยุ่นในการค้นหาข้อมูลเนื่องจากการใช้แผนที่จากกูเกิ้ล แต่อย่างไรก็ตามงานวิจัยนี้ไม่รองรับการสืบค้นของข้อมูลท่องเที่ยวที่อาจมีการเขียนผิดหรือมีรูปแบบการเขียนหรือเรียกที่หลากหลาย ซึ่งต่อมา Bruecker at el. (2008) ได้ออกแบบระบบ ONTO-GIS โดยนำ Canonical Name matching Algorithm มาช่วยในการแก้ปัญหาสำหรับค้นหาชื่อสถานที่ในแผนที่ระบบสารสนเทศทางภูมิศาสตร์ที่สามารถเขียนได้หลากหลายรูปแบบหรือคำที่พ้องรูป พ้องเสียง และใช้หลักการออนโทโลจีเพื่อสนับสนุนการค้นหาในภาษาไทย ภาษาอังกฤษ และภาษาญี่ปุ่น แล้วใช้ Google Map มาช่วยแสดงแผนที่

Park et al. (2009) ได้นำหลักการออนโทโลจีมาใช้ในการออกแบบระบบข้อมูลนักท่องเที่ยวอย่างชาญฉลาด เนื่องจากความก้าวหน้าของระบบเครือข่ายอินเทอร์เน็ตทำให้มีผู้ใช้ผ่านเว็บและมือถือเพิ่มมากขึ้น โดยโดเมนออนโทโลจีประกอบไปด้วยแนวคิดสำหรับศูนย์บริการนักท่องเที่ยว สถานที่ท่องเที่ยวและออนโทโลจีที่ในในระบบ ซึ่งระบบประกอบไปด้วยระบบศูนย์บริการนักท่องเที่ยวที่เก็บออนโทโลจีและฐานข้อมูลแล้วใช้การเชื่อมต่อเชิงความหมายระหว่างออนโทโลจีกับฐานข้อมูล ระบบบริการแผนที่การจราจรแบบให้ข้อมูลแบบทันที ระบบจัดการข้อมูลนักท่องเที่ยว ซึ่งมีการติดต่อกับผู้ใช้แบบเมนูกราฟิกที่สามารถใช้ทั้งเครือข่ายมีสายและไร้สายได้ รวมถึงโทรศัพท์มือถือและPDA อีกทั้งยังใช้ RFID ในบริการข้อมูลแหล่งท่องเที่ยวให้กับนักท่องเที่ยวอีกด้วย แต่ยังพบว่าระบบนี้ไม่รองรับการค้นหาสถานที่ท่องเที่ยวต่าง ๆ และไม่มีระบบวางแผนและแนะนำการท่องเที่ยว ซึ่งเป็นเพียงศูนย์บริการ

ข้อมูลนักเที่ยวเที่ยวเท่านั้น ทำให้ไม่ครอบคลุมความต้องการของนักท่องเที่ยวได้ทั้งหมด ซึ่งระบบบริการวางแผนการท่องเที่ยวนั้นถือว่าเป็นเครื่องช่วยอำนวยวามสะดวก ซึ่งทำให้นักท่องเที่ยวสามารถลดเวลาการสืบค้นข้อมูลท่องเที่ยวและช่วยในการตัดสินใจท่องเที่ยวได้อีกด้วย ซึ่งระบบวางแผนการท่องเที่ยวนั้นมีทั้งที่แสดงผลแบบ Static และแบบ Dynamic เช่น งานวิจัยของ Castillo at el. (2008) ซึ่งได้นำเสนอระบบที่ช่วยวางแผนการท่องเที่ยวให้กับนักท่องเที่ยวแบบไดนามิก โดยสามารถนำพฤติกรรมการท่องเที่ยวในอดีตและลักษณะการเที่ยวที่คล้ายกันของนักท่องเที่ยวมาวิเคราะห์ด้วยหลักการ case-based reasoning อีกทั้งระบบนี้ยังได้นำหลักการออนโทโลจีมาใช้ในการจัดเก็บข้อมูลนักท่องเที่ยว สถานที่ท่องเที่ยว พฤติกรรมการท่องเที่ยว กิจกรรมการท่องเที่ยว การเดินทาง รวมถึงการเดินทางจากที่หนึ่งไปยังอีกที่ด้วยการเดิน โดยระบบการวางแผนท่องเที่ยวนี้สามารถรองรับอุปกรณ์พกพกหรือโทรศัพท์มือถือพร้อมส่งออกแผนที่การเดินทางไปยังอุปกรณ์ดังกล่าวในรูปแบบไฟล์ JPG หรือ PDF โดยใช้ A* Algorithm ในการแนะนำเส้นทาง แต่พบว่ามีข้อเสียคือพบปัญหาในการท่องเที่ยวแบบระบุสถานที่เฉพาะอาจจะไม่สามารถค้นเจอถ้าพิมพ์หรือสะกดไม่ถูกต้อง ซึ่งในงานวิจัยของ Kongthon et al. (2011) ได้พัฒนาระบบสืบค้นข้อมูลท่องเที่ยวที่มีความยืดหยุ่นกว่า เนื่องจากสามารถให้ผู้ใช้ป้อนคำถามที่เป็นประโยคเพื่อสอบถามข้อมูลตามความต้องการในการสืบค้นข้อมูลท่องเที่ยวในประเทศไทยได้ เช่น "รบกวนแนะนำที่พักที่เกาะกูดหน่อยครับไปกัน 10 คน" ซึ่งระบบจะมีกระบวนการตัดคำภาษาไทยโดยเทียบจากพจนานุกรมและจัดกลุ่มของคำที่ตัดได้ให้เข้ารูปแบบโครงสร้างของการจัดเก็บข้อมูลในออนโทโลจีท่องเที่ยวเพื่อส่งรูปแบบโครงสร้างคำเหล่านี้ไปสืบค้นข้อมูลท่องเที่ยวที่ถูกเก็บไว้ในออนโทโลจีท่องเที่ยวและแสดงผลลัพธ์ตามที่ผู้ใช้ได้ทำการสอบถาม แตระบบนี้รองรับแค่ภาษาไทยเท่านั้น เช่นเดียวกันกับงานวิจัยของ Cao et al. (2011) และ Cao and Nguyen (2012) ได้พัฒนาระบบ STAAR (Semantic Tourist information Access and Recommending) สำหรับค้นหาและแนะนำข้อมูลท่องเที่ยวจากการป้อนคำถามแต่ใช้ได้เฉพาะภาษาอังกฤษเท่านั้น โดยใช้ออนโทโลจีในการเก็บข้อมูลท่องเที่ยวและสามารถประมวลผลผ่านโทรศัพท์มือถือด้วยเทคนิคการประมวลผลภาษาธรรมชาติโดยใช้กระบวนการตัดคำและกระบวนการเทียบจากพจนานุกรมหรือคลังคำศัพท์มาใช้ในการสืบค้นข้อมูลตามโครงสร้างคำถามที่ผู้ใช้ต้องการ

จากที่ได้กล่าวาจะเห็นได้ว่าระบบการสืบค้นข้อมูลท่องเที่ยวโยทั่วไปยังไม่รองรับการค้นหาในกรณีที่พิมพ์ชื่อผิดหรือสะกดไม่ถูกต้องตามหลักภาษาไทยหรือภาษาอังกฤษ ซึ่งทำให้ได้ผลลัพธ์ของการค้นหาไม่ตรงกับความต้องการหรือไม่พบสถานที่ที่ต้องการค้นหา อีกทั้งผลลัพธ์ที่ได้ไม่ได้ถูกจัดหมวดหมู่ไว้อย่างเหมาะสม ทำให้เสียเวลาในการคัดเลือกข้อมูลท่องเที่ยวเพื่อให้ได้ข้อมูลท่องเที่ยวหรือกิจกรรมท่องเที่ยวที่น่าสนใจในช่วงเวลานั้นๆ ที่ตรงกับความต้องการทั้งหมดและยังทำให้เกิดความไม่สะดวกในการเข้าถึงข้อมูลท่องเที่ยวในเว็บไซต์เดียว (Panawong et al., 2014) ซึ่งการที่นักท่องเที่ยวจะเที่ยวให้ครบทุกที่ในเวลาอันจำกัดย่อมเป็นไปได้ยาก เพราะแต่ละที่ใช้เวลาในการท่องเที่ยวช้าเร็วไม่เท่ากันขึ้นอยู่กับความสนใจของแต่ละคนและบางสถานที่ท่องเที่ยวยังมีเวลาเปิด ปิดแตกต่างกัน ทำให้นักท่องเที่ยวจะต้องค้นหาข้อมูลเพื่อที่จะวางแผนไว้ล่วงหน้าหรือเลือกเฉพาะสถานที่ท่องเที่ยวที่สนใจเท่านั้น (Sylejmani and Dika, 2011) แม้ว่าการท่องเที่ยวในช่วงเวลาสั้นๆ แต่นักท่องเที่ยวจะค้นหา

ข้อมูลท่องเที่ยวจากแหล่งข้อมูลที่ต่างกันไปหรือโทรศัพท์สอบถามข้อมูลเส้นทาง ของสถานที่ท่องเที่ยวต่าง ๆภายในจังหวัดนั้น ๆ ซึ่งจะใช้เวลามากในการสืบค้นข้อมูลในการตัดสินใจท่องเที่ยว (Maervoet et al.,2008) ทำให้ นฤพนธ์ และ จักรกฤษณ์ (2553) ได้พัฒนาระบบสืบค้นข้อมูลออนโทโลจีท่องเที่ยว โดยใช้ Variation Name Matching Algorithmในการแก้ปัญหาการเขียนชื่อสถานที่หรือคำที่มีการสะกดคล้ายคลึงกันในเว็บไซต์ต่าง ๆหรือที่เกี่ยวข้องสามารถค้นหาเจอได้ครบถ้วนมากที่สุด ซึ่งข้อดีของการ ใช้ Variation Name Matching Algorithm นั้นจะทำให้คำที่เขียนหรือสะกดผิดในเว็บไซต์จะถูกค้นหาเจอในเว็บเดียว แต่อย่างไรก็ตาม วิธีการนี้ทำให้เสียเวลาอย่างมากในกระบวนการค้นหา เนื่องจากมีข้อมูลที่ไม่เกี่ยวข้องของคำที่มีการสะกดผิดพบอยู่เป็นจำนวนมากมาก ทำให้ (นฤพนธ์ และจักรกฤษณ์ 2556) ได้ประยุกต์ใช้อัลกอริทึม ISG ซึ่งเป็น name matching ที่ใช้ในการหาความคล้ายคลึงของตัวอักษรของชื่อ โดย ISG จะทำการกรองข้อมูลที่ไม่เกี่ยวข้องเหล่านั้นก่อนนำไปทำการสืบค้นหาข้อมูลในกูเกิ้ล โดยใช้เทคนิค web crawler เพื่อช่วยลดเวลาในการสืบค้นข้อมูลท่องเที่ยวจากชื่อสถานที่ท่องเที่ยวต่าง ๆได้อย่างมีประสิทธิภาพซึ่งรองรับทั้งภาษาไทยและภาษาอังกฤษ แต่อย่างไรก็ตามผลจากการวัดประสิทธิภาพของการสืบค้นข้อมูลท่องเที่ยวที่มีความหลากหลายนั้น พบว่า ระบบค้นหาชื่อแหล่งท่องเที่ยวทั้งภาษาไทยและภาษาอังกฤษมีค่าประสิทธิภาพใกล้เคียงกันซึ่งอยู่ในเกณฑ์พอใช้ คิดเป็น 71.21% และ 70.49% ตามลำดับ และไม่ได้มีการนำเอา อัลกอริทึม name matching อื่น ๆ มาใช้และเปรียบเทียบกันในระบบสืบค้นข้อมูลท่องเที่ยว

กรอบแนวความคิด

การวิเราะห์ประสิทธิภาพและอัลกอริทึมเนมแมทชิ่งมีขั้นตอนการดำเนินงานขั้นตอนดังนี้

- พัฒนาเนมแมทชิ่งอัลกอริทึมของกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง
- ทดสอบเนมแมทชิ่งอัลกอริทึมและทำการวัดประสิทธิภาพของการสืบค้นของเนมแมทชิ่งอัลกอริทึม โดยใช้วิธีการ F-Measure โดยใช้ฐานข้อมูลชื่อจังหวัดที่มีการเขียนผิดหรือสะกดผิดในกูเกิ้ล ซึ่งมีสูตรการคำนวณประสิทธิภาพมีดังต่อไปนี้

$$f = 2 * (\frac{P * R}{P + R}) \quad (1)$$

$$P = \frac{|TP|}{(|TP| + |FP|)} \quad (2)$$

$$R = \frac{|TP|}{(|TP| + |FN|)} \quad (3)$$

โดยที่	f	คือ	ค่า f-score
	P	คือ	ค่า Precision
	R	คือ	ค่า Recall
	TP	คือ	จำนวนชื่อที่ควรจะ Match และ Match (True Positive)
	FP	คือ	จำนวนชื่อที่ควรจะไม่ Match แต่ Match (False Positive)
	FN	คือ	จำนวนชื่อที่ควรจะ Match แต่ไม่ Match (False Negative)

Name Matching Algorithm

Name Matching Algorithm เป็นอัลกอริทึมที่ใช้ในการเปรียบเทียบชื่อสองชื่อว่ามีความคล้ายกันหรือไม่ (Match/No match) ซึ่งได้มีการแบ่งชนิดของName Matching Algorithmออกเป็น 4 ประเภท (Snae and Brueckner, 2006) ได้แก่

- กฎพื้นฐานของตัวสะกด (Spelling/string analysis based algorithms)ใช้ในการเปรียบเทียบความคล้ายคลึงกันระหว่างตัวอักษรของชื่อ 2 ชื่อ ว่ามีหลักการเขียนเหมือนกันอย่างไร และผลของการเปรียบเทียบนี้จะออกมาในรูปว่าคล้ายกันหรือไม่ (Match/No match) ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Guthและ Levenshtein
- กฎพื้นฐานของการออกเสียง (Phonetic/sound base algorithms) เป็นการใช้โครงสร้างของเสียงเป็นตัวเปรียบเทียบค่าความคล้ายคลึงกัน ซึ่งใช้ในการแก้ปัญหาชื่อที่มีเสียงเดียวกันแต่เขียนหรือสะกดต่างกัน โดยใช้วิธีการจัดกลุ่มตัวอักษรจัดให้เสียงเดียวกันอยู่ในกลุ่มตัวอักษรเดียวกัน ตัวอย่างเช่น ตัวอักษร Phใช้จัดอยู่ในกลุ่มตัวอักษร F หรือในภาษาไทย "จ ท ธ ต ฏ" ให้จัดอยู่ในกลุ่มแม่กด (ด) อาทิเช่น คำว่า กาจ กับ กาด จะออกเสียงเหมือนกัน ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Soundex, Soundex2, Metaphone, NYSIIS และ Phonex
- วิธีการรวมกัน (Composite methods) เป็นวิธีการรวมกันระหว่างกฎพื้นฐานของตัวสะกด (Spelling/string analysis based algorithms) และกฎพื้นฐานของการออกเสียง (Phonetic/sound base algorithms) ซึ่งการรวม 2 วิธีนี้เป็นการแก้ปัญหาของชื่อเกี่ยวกับคำพ้องรูปพ้องเสียงหรือคำที่เขียนเหมือนกันแต่ออกเสียงต่างกันและคำที่เขียนต่างกันแต่ออกเสียงเหมือนกัน ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ Simplex และ ISG
- วิธีการไฮบริด (Hybrid approaches)เป็นการใช้ค่าความน่าจะเป็นในการวัดค่าความคล้ายคลึงของชื่อ (Snae and Diaz, 2002) ซึ่งอัลกอริทึมประเภทนี้ ได้แก่ LIG1, LIG2, LIG3 (ย่อมากจาก Levenshtein Index of Similarity and Guth)

ในงานวิจัยนี้จะวัดประสิทธิภาพของอัลกอริทึมเนมแมชซึ่งในกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียงเท่านั้นซึ่งเนมแมทชิ่งอัลกอริทึมในกลุ่มนี้เป็นการเปรียบเทียบระหว่างรหัสเสียงของชื่อ โดยการนำรหัสเสียงของชื่อ 2 ชื่อมาเปรียบเทียบกันโดยตรง ถ้าเหมือนกันแสดงว่าชื่อทั้ง 2 ชื่อ Match กัน แต่ถ้า

ไม่เหมือนกันแสดงว่าชื่อทั้ง 2 ชื่อ No Match กัน ซึ่งเนมแมทชิ่งอัลกอริทึมในกลุ่มนี้มี 6 อัลกอริทึม ได้แก่ Soundex, Soundex2, Phonex, Metaphone, Simplex และ NYSIIS

1. Soundex

Soundex Algorithm ถูกออกแบบมาใช้สำหรับชื่อภาษาอังกฤษ (Winchester, 1970) Soundex Algorithm ได้ถูกนำไปใช้ เพื่อลดจำนวนระเบียนของการแต่งงานโดยการเข้ารหัสทั้งนามสกุลของสามีและนามสกุลเดิมของภรรยา และยังมีการนำอัลกอริทึมนี้ ไปใช้ในการเชื่อมโยงประชากรอีกด้วย แต่อย่างไรก็ตาม จะพบว่าการละเว้นการออกเสียงอาจจะทำให้ข้อมูลที่สมควร Match กลับได้ผลเป็นไม่ Match และข้อมูลที่ไม่ Match กลับให้ผลที่ Match กัน (Davey And Jarvis, 1990) ซึ่งข้อเสียที่พบจากการใช้ Soundex Algorithm คือ ไม่เหมาะกับชื่อที่สั้น หรือชื่อที่มีจำนวนสระมาก (Gill et, al., 1993)

Soundex Algorithm จะทำการเปลี่ยนชื่อเป็นรหัสของ Soundexซึ่งประกอบด้วย 1 ตัวอักษร และ 3 ตัวเลข โดยมีขั้นตอนการเปลี่ยนดังต่อไปนี้

1) เก็บตัวอักษรตัวแรกของ Input ชื่อไว้
2) ถ้าตัวอักษร A, E, H, I, O, U, W, Y ไม่ใช่ตัวอักษรตัวแรกให้ตัดออกไป
3) จากข้อที่ 1 เปลี่ยนอักษรที่เหลือเป็นรหัส Soundexดังแสดงในตารางที่ 1
4) ผลลัพธ์ของรหัส Soundexต้องอยู่ในรูปของตัวอักษร 1 ตัวและตามด้วยตัวเลข 3 ตัว
5) จากข้อ 4 ถ้าผลลัพธ์ของรหัส Soundexน้อยกว่า 4 ตัวรหัส Soundexให้ใส่ศูนย์แทนจนครบรหัส แต่ถ้ารหัส Soundexมากกว่า 4 รหัสให้ตัดออก

ตารางที่ 1 แสดงการเปลี่ยนอักษรเป็นรหัส Soundex

Letter	Number is coded to
B, P, F and V	1
C, G, J, K, Q, S, X and Z	2
D and T	3
L	4
M and N	5
R	6

2. Soundex 2

Soundex 2 ได้ปรับปรุงมาจากอัลกอริทึม Soundex ซึ่งจะมีขั้นตอนเหมือนกับ Soundex Algorithm แต่จะมีการเปลี่ยนเพียงขั้นตอนที่ 3 โดยรหัสของ Soundex 2 สามารถแสดงในตารางที่ 2

ตารางที่ 2 แสดงการเปลี่ยนอักษรเป็นรหัส Soundex 2

Letter	Number is coded to
B and P	1
F and V	2
C, K and S	3
G and J	4
Q, X and Z	5
D and T	6
L	7
M and N	8
R	9

3. Metaphone

ส่วนใหญ่จะใช้แก้ปัญหากับชื่ออังกฤษ โดยอัลกอริทึมมีขั้นตอนดังต่อไปนี้

1) ถ้ามีตัวเลขให้ตัดตัวเลขออก แล้วเปลี่ยนอักษรทุกตัวให้เป็นตัวใหญ่
2) ถ้า KN หรือ GN หรือ PM หรือ AE หรือ WR เป็นตัวแรกให้วางไว้ตำแหน่งแรก
3) ถ้า X เป็นตัวอักษรแรกให้เปลี่ยนเป็น S
4) ถ้า WH เป็นตัวอักษรแรกให้เปลี่ยนเป็น W
5) ถ้ามีอักษรที่เหมือนกันไม่ต้องเพิ่มรหัส
6) ถ้าตัวอักษร A, E, I, O, U และ Y ไม่ใช่ตัวอักษรตัวแรกให้ตัดออกไป
7) เปลี่ยนตัวอักษรให้เป็นรหัสของ Metaphoneดังแสดงในตารางที่ 3

ตารางที่ 3 แสดงการเปลี่ยนตัวอักษรเป็นรหัส Metaphone

ตัวอักษร	รหัส	เงื่อนไข
B	B	ยกเว้นถ้าอยู่ตำแหน่งสุดท้ายของคำหลังจากตัวอักษร "m"
C	X	(เสียง 'sh') ถ้า "-cia-" หรือ "-ch-"
	S	เป็น "-ci-", "-ce-" หรือ "-cy-"
		ถ้าไม่ออกเสียง "-sci-", "-sce-" หรือ "-scy-"

ตัวอักษร	รหัส	เงื่อนไข
	K	มิฉะนั้น, รวมทั้ง "-sch-"
D	J	ถ้าอยู่ใน "-dge-", "-dgy-", "-dgi"
	T	นอกนั้นเป็น
F	F	
G		ไม่ออกเสียงถ้ามันอยู่ใน "-gh-" และไม่อยู่ในตำแหน่งสุดท้าย หรือก่อนหน้าสระ
		ใน "-gn" หรือ "-gned"
		ใน "-dge-", อื่นๆ., ที่เหนือกว่ากฎ
	J	ถ้า "i", หรือ "e", หรือ "y" ก่อน, ถ้าไม่อยู่เป็นคู่ติดกัน "gg"
	K	นอกนั้นเป็น
H		ไม่ออกเสียงถ้าอยู่หลังสระและไม่มีสระตามมา
	H	นอกนั้นเป็น
J	J	
K		ถ้าไม่ออกเสียงใน "c"
	K	นอกนั้นเป็น
L	L	
M	M	
N	N	
P	F	ถ้า "h" ก่อน
	P	นอกนั้นเป็น
Q	K	
R	R	
S	X	(sh) ถ้า "h" ก่อน หรือใน "-sio-" หรือ "-sia-"
	S	นอกนั้นเป็น
T	X	(sh) ถ้า "h" ก่อน หรือใน "-sio-" หรือ "-sia-"
	0 (Zero)	(th) ถ้า "h" ก่อน
		ถ้าไม่ออกเสียงใน "tch-"
	T	นอกนั้นเป็น
V	F	
W		ไม่ออกเสียงถ้าไม่ถูกตามโดยสระ
	W	ถ้าถูกตามโดยสระ
X	KS	
Y		ไม่ออกเสียงถ้าไม่ถูกตามโดยสระ
	Y	ถ้าถูกตามโดยสระ
Z	S	

4. Phonex

เกิดจากการรวมกันของสองกระบวนการ คือ Soundexและ Metaphone ซึ่งกระบวนการนี้พบว่า สามารถทำงานได้ดีเมื่อนำไปใช้กับชื่อในภาษาอังกฤษ (Lait and Randell, 1998)

ขั้นตอนของ Phonexในการแปลงชื่อให้เป็นรหัส 4 ตัวอักษร มีดังต่อไปนี้

1) ลบตัวอักษร S ที่ต่อท้ายออก
2) ถ้ามี KN หรือ WR ขึ้นต้นให้ตัดตัวอักษรแรกทิ้งไป
3) ถ้ามีตัวอักษรดังต่อไปนี้ขึ้นต้นให้เปลี่ยนดังนี้
4) ตัวอักษร H ให้ตัดทิ้ง
5) ตัวอักษร E I O U Y ให้เปลี่ยนเป็น A
6) ตัวอักษร K และ Q ให้เปลี่ยนเป็น C
7) ตัวอักษร P ให้เปลี่ยนเป็น B
8) ตัวอักษร J ให้เปลี่ยนเป็น G
9) ตัว PH และ V ให้เปลี่ยนเป็น F
10) ตัว Z ให้เปลี่ยนเป็น S
11) ให้เก็บตัวอักษรตัวแรกไว้แล้วลบตัวอักษร A, E, H, I, O, U, W, Y ในตำแหน่งอื่นออก
12) กำหนดหมายเลขดังต่อไปนี้ให้ตัวอักษรที่เหลือหลังตัวแรก
13) B, F, P, V เปลี่ยนเป็น 1
14) C, G, J, K, Q, S, X, Z เปลี่ยนเป็น 2
15) D, T เปลี่ยนเป็น 3 ถ้าไม่ตามด้วย C
16) L เปลี่ยนเป็น 4 ถ้าไม่ตามด้วยสระและเป็นตัวอักษรสุดท้าย
17) M, N เปลี่ยนเป็น 5 ข้ามไปพิจารณาตัวอักษรถัดไปถ้าตัวอักษรถัดไปเป็น D หรือ G
18) R เปลี่ยนเป็น 6 ถ้าไม่ตามด้วยสระและเป็นตัวอักษรสุดท้าย
19) ข้ามตัวอักษรปัจจุบันถ้ามีหมายเลขเดียวกับตัวอักษรก่อนหน้า
20) รหัสของ Phonexต้องมีลักษณะ คือ ตัวอักษรตามด้วยตัวเลข 3 ตัว
21) ถ้ารหัสมีตัวเลขไม่ถึง 3 ตัวให้เติมด้วย 0 จนครบ และถ้ามีตัวเลขครบ 3 ตัวแล้วยังมีตัวอักษรที่ยังไม่ได้เปลี่ยนให้เปลี่ยนที่ตัวเลขสุดท้าย

5. NYSIIS

เป็นวิธีการทางตัวอักษรที่ง่ายและให้ผลรหัสอัลกอริทึมเหมือนกับ Soundex อย่างไรก็ตาม NYSIIS ต่างจาก Soundexตรงที่มันเก็บข้อมูลเกี่ยวกับตำแหน่งของสระในการเข้ารหัสของคำโดยแปลงสระทั้งหมดให้เป็นตัวอักษร A (Gill et al., 1993) โดยวิธีการเข้ารหัสเบื้องต้นของ NYSIIS มีดังนี้

1) ใช้ตัวอักษรตัวแรกของชื่อมาเป็นรหัสตัวแรกของ NYSIIS
2) ลบสระที่ซ้ำกันให้เหลือเพียงหนึ่งตัว
3) ถ้าชื่อประกอบด้วยสระครบทุกตัว (AEIOU) ให้ตัดสระทิ้ง
4) แทนค่าสระด้วยตัว A
5) ถ้าตัวอักษรตัวสุดท้ายเป็นสระ ให้ตัดทิ้ง
6) ตัวอักษรที่เหลือให้เปลี่ยนเป็นรหัส NYSIIS

6. Simplex

เป็นวิธีที่มีการปรับปรุงจากวิธีของ NYSIIS และ Soundex Algorithm ซึ่งจะเพิ่มจำนวนการ Match มากขึ้นจากเดิม โดยการแก้ไขส่วนใหญ่เน้นไปที่การแก้ไขตัวอักษร การเข้ารหัสของ Simplex มีพื้นฐานมาจาก NYSIIS โดยการเปลี่ยนสระ "AIEOUY" เป็น A ทั้งหมด

1) ทำตามขั้นตอนข้อ 1-6 ของ NYSIIS Algorithm
2) เข้ารหัสตัวอักษรตัวสุดท้ายและตัวอักษรตัวอื่นๆ ดังต่อไปนี้
3) สำหรับตัวอักษรตัวสุดท้าย เข้ารหัสดังนี้
4) เปลี่ยน X เป็น C
5) เปลี่ยน V เป็น F
6) เปลี่ยน SE, ES, AS, CE เป็น S
7) เปลี่ยน YE, EE, IE, EY, EI เป็น Y
8) เปลี่ยน NT, ND เป็น N
9) สำหรับตัวอักษรตัวอื่นๆ เข้ารหัสดังนี้
10) เปลี่ยน PH เป็น F
11) เปลี่ยน JH เป็น J
12) เปลี่ยน HN เป็น N
13) เปลี่ยน TH เป็น T
14) เปลี่ยน RH, RM เป็น R
15) เปลี่ยน WR เป็น R
16) เปลี่ยน WM, WN, M เป็น N

17) ถ้ามี K ตามด้วย N ให้เปลี่ยนเป็น N แต่ถ้า K ตามด้วยตัวอักษรอื่นที่ไม่ใช่ N ให้เปลี่ยนเป็นตัว C

18) ถ้ามี Z เป็นตัวอักษรตัวสุดท้ายให้เปลี่ยน Z เป็น S แต่ถ้าไม่ใช่ Z เป็นตัวสุดท้าย ให้คงตัวอักษรเดิมไว้

19) ถ้ามี Y เป็นทั้งตัวอักษรตัวแรกและตัวสุดท้าย ให้คงเดิม แต่ถ้าเป็นตัวอักษรอื่นให้เปลี่ยนเป็น A

ระเบียบวิธีการวิจัย

ในหัวข้อนี้จะอธิบายขั้นตอนการวิเคราะห์ประสิทธิภาพของอัลกอริทึมเนมแมชชิ่งซึ่งสามารถแบ่งออกเป็น 3 ขั้นตอน หลัก ดังต่อไปนี้

- การสร้างคำค้นหาหรือชื่อของจังหวัด ที่สามารถเขียนได้หลากหลายรูปแบบในภาษาอังกฤษ
- คัดกรองและจัดเก็บข้อมูลชื่อเพื่อใช้ในการวัดประสิทธิภาพของอัลกอริทึมเนมแมชชิ่ง
- วัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure

การสร้างคำค้นหาหรือชื่อของจังหวัด

ทำการป้อนชื่อจังหวัดที่เขียนถูกต้องเป็นภาษาอังกฤษ จากนั้นทำการสร้างชื่อที่สามารถเขียนได้หลากหลายหรือคล้ายคลึงกันกับชื่อที่เขียนถูกต้องโดยใช้อัลกอริทึม name variation finding ซึ่งจะทำการสร้างชื่อจากความคล้ายคลึงของพยัญชนะและตัวสะกด (Snae et al., 2007; Snae and Brueckner, 2009) ดังแสดงตัวอย่างตามตารางที่ 4

ตารางที่ 4 แสดงการทำงานของอัลกอริทึมของ name variation finding สำหรับการสร้างคำคล้ายคลึงของ phitsanulok

คำที่ใช้	ตำแหน่งตัวอักษร	ตัวอักษรที่เปลี่ยน	เปลี่ยนเป็น	คำที่ได้
Phitsanulok	4	t	ตัดออก	Phisanulok
Phitsanulok	2	h	ตัดออก	Pitsanulok
Phitsanulok	2,4	h,t	ตัดออก	Pisanulok
Phitsanulok	9	l	r	Phitsanurok
Phitsanulok	4	t	ตัด ออก เพิ่ม k	Phisanuklok
Phitsanulok	4,5	t,s	ตัด ออก เปลี่ยนs-> z	Phizanulok
Phitsanulok	8	u	oo	Phitsanoolok
Phitsanulok	8	u	ok	phitsanoklok

จากตารางที่ 4 จะเห็นได้ว่าคำว่า Phitsanulok สามารถเขียนได้หลากหลายรูปแบบ เช่น Phisanulok (t ถูกตัดออกในตำแหน่งที่ 4) หรือ Pitsanulok (h ตำแหน่งที่ 2 ถูกตัดออก) เป็นต้น ซึ่งในงานวิจัยนี้จะใช้ตัวอย่างชื่อที่เขียนเป็นคำภาษาอังกฤษของจังหวัดในประเทศไทย 20 จังหวัด แต่ละจังหวัดประกอบด้วยชื่อที่สามารถเขียนได้หลากหลายที่ค้นเจอในกูเกิ้ลและมีข้อมูลท่องเที่ยวปรากฏอยู่จริง จำนวน 100 ชื่อ (2,000 ชื่อ) มาใช้ในการวัดประสิทธิภาพของ อัลกอริทึมเนมแมชชิ่ง

คัดกรองและจัดเก็บข้อมูลชื่อ

นำชื่อที่เขียนได้หลายรูปแบบทั้ง 20 จังหวัด ที่ได้จาก name variation finding ไปทำการค้นหาข้อมูลท่องเที่ยวใน กูเกิ้ล และทำการคัดกรองเฉพาะชื่อแต่ละจังหวัดที่มีข้อมูลท่องเที่ยวปรากฏอยู่จริง (รูปที่ 1) ที่ค้นเจอในกูเกิ้ล จำนวน 100 ชื่อ (2,000 ชื่อ) และจัดเก็บชื่อจังหวัดและชื่อที่มีความคล้ายคลึงกันเหล่านี้ลงฐานข้อมูล

Name Variation of		
No.	Name Variation	Records were from Google
1	phitsanulok	5,110,000 records
2	phisanulok	186,000 records
3	pitsanulok	32,800 records
4	pisanulok	5,510 records
5	phitsanurok	1,400 records
6	phisanuklok	134 records
7	phizanulok	5 records
8	phitsanoolok	5 records
9	phitsanoklok	4 records

รูปที่ 1 แสดงตัวอย่างจำนวนรายการที่ค้นพบโดยกูเกิ้ลของคำค้นหา พิษณุโลก (Phitsanulok)

จากนั้นนำรายชื่อที่ได้ทั้งหมดมาไปเข้ากระบวนการอัลกอริทึมเนมชิ่งแต่ละวิธี (ดูหัวข้อ กรอบแนวคิดของ name maching algorithm) จากนั้นทำการวิเคราะห์ประสิทธิภาพด้วยการคำนวณหาประสิทธิภาพของแต่ละอัลกอริทึมของ อัลกอริทึมเนมแมชชิ่งด้วย F-Measure

วัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure

ขั้นตอนการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure สำหรับกลุ่มอักอริทึมเนมแมชชิ่งที่มีการเปลี่ยนชื่อเป็นรหัสเสียงนั้น สามารถแสดงวิธีการคำนวณค่าประสิทธิภาพโดยใช้ตัวอย่างของอัลกอริทึม Soundex ดังตารางที่ 5

ตารางที่ 5 แสดงตัวอย่างการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure ของ อัลกอริทึม Soundex

ชื่อจังหวัดที่ถูกต้อง และ Soudex code	ฐานข้อมูลชื่อที่มีความคล้ายคลึงกัน	Soundex code	Result (match/no match)	Match type (TP, FP, FN)
Phitsanulok	Phisanulok	P254	No match	FN
P325	Pitsanulok	P325	Match	TP
	Pisanulok	P254	No match	FN
	Phitsanurok	P325	Match	TP
	Phisanuklok	P252	No match	FN
	Phizanulok	P254	No match	FN
	Phitsanoolok	P325	Match	TP
	phitsanoklok	P325	Match	TP
	...	...	...	
	Phitjint	P325	Match	FP

จากตารางที่ 5 สามารถคำนวณหาค่าต่างๆ ได้ดังต่อไปนี้

- หาค่า TP ได้จากการนับชื่อที่ Match ในการเปรียบเทียบ Soundex Code ของชื่อจังหวัดที่เขียนถูกต้อง Phitsanulok (P325) กับชื่อจังหวัดในฐานข้อมูล โดยในตัวอย่างนี้ TPเท่ากับ 4
- หาค่า FN ได้จากการนับชื่อที่ No match ในการเปรียบเทียบ Soundex Code ของชื่อจังหวัดที่เขียนถูกต้อง Phitsanulok (P325) กับชื่อจังหวัดในฐานข้อมูล โดยในตัวอย่างนี้ TPเท่ากับ 4
- หาค่า FP ได้จากการนับชื่อที่ Match ในการเปรียบเทียบ Soundex Code ของชื่อจังหวัดที่เขียนถูกต้อง Phitsanulok (P325) กับชื่อจังหวัดในฐานข้อมูล โดยในตัวอย่างนี้ TP เท่ากับ 1
- คำนวณหาค่า Precision, ค่า Recallและค่า F-Score ของ จังหวัด Phitsanulok สามารถหาค่าต่าง ๆ ได้ดังนี้
  - Precision : 4/(4+1)*100 = 80 %
  - Recall : 4/(4+4)*100 = 50%
  - F-Score : 2*(80*50)/(80+50) = 61.54%

ดังนั้นสรุปได้ว่าค่า ประสิทธิภาพของ อัลกอริทึม soundex สำหรับจังหวัด phitsanulok มีค่าเท่ากับ 61.54% จากนั้นทำการหาค่า f-score จนครบทุก 20 จังหวัดในฐานข้อมูล และครบทุกอัลกอริทึมเนม แมชชิ่ง พร้อมทั้งสรุปผลการวิเคราะห์ประสิทธิภาพ

ผลการวิจัย

ผลจากการวิเคราะห์ประสิทธิภาพของอัลกอริทึมเนมแมชชิ่ง 6 วิธีการโดยใช้ ข้อมูลคำของจังหวัด 20 จังหวัด และคำที่คล้ายคลึงที่ปรากฏอยู่จริงจากการสืบค้นข้อมูลท่องเที่ยวจาก กูเกิ้ลนั้น สามารถแสดงคาประสิทธิภาพดัง ตารางที่ 6 และ 7

ตารางที่ 6 แสดงผลการวัดประสิทธิภาพของอัลกอริทึม Soundex ด้วย F-Measure (ภาษาอังกฤษ) จำนวน 20 จังหวัด

Basename(b)	P	R	f-score	%
buriram	1.000	0.8	0.8889	88.89
kalasin	1.000	0.82	0.9011	90.11
nakhonratchasima	1.000	0.39	0.5612	56.12
mahasarakham	0.523	0.575	0.5478	54.78
ubonratchathani	0.816	0.488	0.6107	61.07
phichit	0.929	0.898	0.9132	91.32
phitsanulok	0.942	0.857	0.8975	89.75
pathumthani	0.652	0.545	0.5937	59.37
phetchabun	0.763	0.738	0.7503	75.03
maehongson	0.629	0.667	0.6474	64.74
nakhonsawan	0.568	0.917	0.7015	70.15
chiangmai	0.965	0.902	0.9324	93.24
sukhothai	0.885	0.963	0.9224	92.24
chiangrai	**0.307**	**0.964**	**0.4657**	**46.57**
kamphaengphet	**1.000**	**0.7**	**0.8235**	**82.35**

Basename (b)	P	R	f-score	%
udonthani	1.000	0.722	0.8386	83.86
nakhonsithammarat	0.477	0.7	0.5674	56.74
ayutthaya	0.750	0.98	0.8497	84.97
suphanburi	1.000	0.32	0.4848	48.48
sakonnakhon	0.649	0.6	0.6235	62.35
Average				77.09

จากการวัดประสิทธิภาพของอัลกอริทึมเนมแมชชิ่งด้วย F-measure นั้นพบว่า ค่าF-Score หรือประสิทธิภาพในการจับคู่ชื่อที่ให้ค่าที่สูงที่สุดสำหรับระบบสืบค้นข้อมูลท่องเที่ยว คือ Soudex (77.09%) รองลงมาคือ Soudex2 (75.68%) และ Phonex (74.34%) ตามลำดับ ซึ่งจเห็นว่ามีค่าใกล้เคียงกันและถือว่ามีประสิทธิภาพของระบบอยู่ในเกณฑ์ดี โดยผลการวัดประสิทธิภาพของอัลกอริทึมทั้งหมดแสดงในตารางที่ 7

ตารางที่ 7 แสดงผลการวัดประสิทธิภาพเนมแมทชิ่งอัลกอริทึมด้วยวิธีการ F-measure

Algorithm	F- Score (%)
Soundex	77.09
Soundex2	75.68
Metaphone	62.32
Phonex	74.34
Simplex	59.55
NYSIIS	52.09

อภิปรายผลการวิจัย

ในงานวิจัยนี้ผู้วิจัยได้นำเสนอผลของการวิเคราะห์ประสิทธิภาพของระบบสืบค้นข้อมูลท่องเที่ยวโดยใช้อัลกอริทึม variation name finding มาช่วยในคัดเลือกชื่อจังหวัดที่มีความคล้ายคลึง

หรือใกล้เคียงกับชื่อจังหวัดที่เขียนถูกต้องเพื่อนำไปใช้เป็นข้อมูลในการเปรียบเทียบอัลกอริทึมเนมแมชซิ่งแต่ละวิธี จากผลการทดสอบการวัดประสิทธิภาพและเปรียบเทียบอัลกอริทึมเนมแมทชิ่งสรุปได้ว่าเนมแมทชิ่งอัลกอริทึมแต่ละวิธีมีความสามารถเฉพาะแต่ละด้านไม่เหมือนกัน ซึ่งการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมเพื่อให้ทราบว่าเนมแมทชิ่งอัลกอริทึมวิธีไหนให้ประสิทธิภาพดีที่สุดสำหรับการสืบค้นชื่อหรือข้อมูลท่องเที่ยวภาษาอังกฤษ ซึ่งเนมแมทชิ่งอัลกอริทึมที่มีความสามารถในการสืบค้นข้อมูลท่องเที่ยวได้ค่อนข้างดีที่สุดในจำนวนเนมแมทชิ่งอัลกอริทึมของกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียง ได้แก่ Soudex, Soudex2 และ Phonex ตามลำดับ ซึ่งระบบสืบค้นข้อมูลท่องเที่ยวนี้สามารถนำอัลกอริทึมทั้ง 3 วิธี ไปประยุกต์ใช้โดยจะช่วยให้ข้อมูลท่องเที่ยวที่มีการเขียนผิดรูปผิดเสียงสามารถค้นหาเจอได้ในอินเทอร์เน็ตได้อย่างรวดเร็ว แต่อย่างไรก็ตามควรมีการเพิ่มจำนวนข้อมูลชื่อสถานที่ท่องเที่ยวที่นำมาทดสอบหาประสิทธิภาพให้มีความหลากหลายรูปแบบให้ครอบคลุมข้อมูลท่องเที่ยวมากขึ้นเพื่อผลลัพธ์ที่ได้จะมีความน่าเชื่อถือยิ่งขึ้นไปด้วย

ข้อเสนอแนะ

จากการวัดประสิทธิภาพและเปรียบเทียบเนมแมทชิ่งอัลกอริทึมของกลุ่มที่มีการเปลี่ยนชื่อเป็นรหัสเสียงทั้ง 6 อัลกอริทึมจะเห็นได้ว่าผลการทดสอบที่ได้มีความน่าเชื่อถือในระดับค่อนข้างดี แต่อย่างไรก็ตามข้อเสนอแนะที่ควรมีการปรับปรุงให้ดียิ่งขึ้น มีดังต่อไปนี้

1) ควรทำการวิเคราะห์ประสิทธิภาพของอัลกอริทึมเนมแมชซิ่งของกลุ่มที่มีการเปรียบเทียบกับค่าThreshold เพิ่มอีกด้วย ซึ่งเนมแมทชิ่งอัลกอริทึมในกลุ่มนี้มี 6 อัลกอริทึมได้แก่ Guth, Levenshtein, ISG, LIG1, LIG2 และ LIG3
2) ควรมีการวัดประสิทธิภาพในภาษาอื่น ๆ นอกเหนือจากภาษาอังกฤษเพิ่มมากขึ้นเช่น ภาษาไทย ซึ่งการประยุกต์ใช้เนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทยนั้น Snae and Bruckner (2009) ได้พัฒนาเนมแมทชิ่งอัลกอริทึมสำหรับภาษาไทย คือ Metasound ซึ่งนำไปใช้กับการตั้งชื่อและเปลี่ยนชื่อไทยให้สอดคล้องกับชื่อบิดาและมารดา หรือเปลี่ยนชื่อให้มีความคล้ายคลึงกับชื่อเดิม
3) ควรมีการเพิ่มการวัดประสิทธิภาพด้วยวิธีการอื่นเพิ่มมากขึ้นเช่น หลักการของ Lati และ Randell (1998) และ หลักการจัดกลุ่ม เป็นต้น
4) ควรมีการทดลองเพื่อแสดงผลการเปรียบเทียบเวลาในการประมวลผลเนมแมทชิ่งอัลกอริทึมเพื่อเป็นส่วนหนึ่งในการเลือกเนมแมทชิ่งอัลกอริทึมในการนำไปประยุกต์ในการสืบค้นและเพื่อเพิ่มความรวดเร็วของระบบสบค้นข้อมูลท่องเที่ยวอีกด้วย
5) ควรใช้เพิ่มจำนวนฐานข้อมูลของชื่อ ให้ครบทุกจังหวัดที่มีข้อมูลท่องเที่ยวปรากฏ จะทำให้ค่าประสิทธิภาพมีความเชื่อถือได้มากยิ่งขึ้น

รายการอ้างอิง/References

นฤพนธ์ พนาวงศ์ และจักรกฤษณ์ เสน่ห์. 2554. การพัฒนาระบบสืบค้นข้อมูลออนโทโลจีท่องเที่ยวด้วยภาษา SPARQL. **การประชุมทางวิชาการนเรศวรวิจัย ครั้งที่ 7**, 195-204.

นฤพนธ์ พนาวงศ์ และ จักรกฤษณ์ เสน่ห์. 2553. ระบบค้นหาสถานที่ท่องเที่ยวในประเทศไทยด้วยหลักการออนโทโลจีและเนมแมทชิ่ง. วารสารวิชาการทางด้านเทคโนโลยีสารสนเทศ, ปีที่ 1 ฉบับที่ 2. 60-69.

นฤพนธ์ พนาวงศ์ และจักรกฤษณ์ เสน่ห์ นมะหุต. 2556. การวิเคราะห์ประสิทธิภาพของระบบสืบค้นข้อมูลออนโทโลจีท่องเที่ยวด้วยอัลกอริทึม ISG และ Name Variation Matching. **วารสารวิทยาศาสตร์ มหาวิทยาลัยนเรศวร**, ปีที่ ฉบับที่ .92. 47-64.

สุรางค์รัตน์ เชาว์โคกสูง และ จักรกฤษณ์ เสน่ห์. 2552. A Web-Based Geographic Information System & Tourist Information System by Animation, **การประชุมวิชาการ เทคโนโลยีอวกาศและภูมิสารสนเทศแห่งชาติ .**

Brückner, M., Snae, C. and Payakpate, J. 2008. Ontology-based name matching of toponyms for Geographical Information Systems (ONTO-GIS). **IADIS International Conference ICT, Society, and Human Beings**. 108-114.

Cao, T., Nguyen, Q., Nguyen, A. and Le, T. 2011. Integrating open data and generating travel itinerary in semantic-aware tourist information system. **In proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services**. 214-221.

Cao, T. and Nguyen, Q. 2012. Semantic approach to travel information search and Itinerary recommendation. International Journal of Web Information Systems. Vol 8. No.3, 256-277.

Castillo, L., Armengol, E., Onaindia, E., Sebastia, L., Gonzalez-Boticario, J., Rodriguez, A., Fernandez, S., Arias, J.D., and Borrajo, D. 2008. SAMAP: An user-oriented adaptive system for planning tourist visits. **Expert Systems with Applications**. Volume 34. No. 2. 1318-1332.

Davey, C. and Javis, A.S. (1990). Microcomputers for Microhistory: a database approach to the construction of small English populations. History and Computing, Vol 2. No.3. 187-193.

Gill, L.E., Goldacre, M.J., Simmons, H.M., Bettley, G.A. and Griffith, M. (1993).Computerised Linkage of Medical Records: Methodological Guidelines.Journal of Epidemiology and Community Health, Vol .47. 316-319.

Kongthon, A., Kongyoung, S., Haruechaiyasak, C. and Palingoon, P. 2011. A Semantic Based Question Answering System for Thailand Tourism Information. **In proceedings of the KRAQ11 Workshop: Knowledge and Reasoning for Answering Questions.** 38-42.

Lait, A. J. and Randell, B. 1998. An Assessment of Name Matching Algorithm. Society of Indexers Genealogical Group, Newsletter Contents. SIGGNL issues 17. 1-31.

Maervoet, J., Souffriau, W., Berghe, G. V. and Oudheusden, D. V. 2008. Tourist Decision Support for Mobile Navigation Systems: a Demonstration, **Applied Artificial Intelligence.** Vol. 22. No. 10. 964-985

Panawong, N., Snae Namahoot, C. and Michael Brueckner. 2014. Classification of Tourism Web with Modified Naïve Bayes Algorithm. **Advanced Materials Research** . Vol. 931-932, 1360-1364.

Park, H., Kwon, S. and Kwon, H. C. 2009. Ontology-based Approach to Intelligent Ubiquitous Tourist Information System. Proceedings of the 4th International Conference on Ubiquitous Information Technologies & Applications.1-6.

Snae, C. and Brueckner, M. 2006. Concept and Rule Based Naming System. **The Journal of Issues in Informing Science and Information Technology**, 3, 619-634.

Snae, C. and Brueckner, M. 2007. LOWCOST: Local organization search with consolidated ontologies for name, space and time. **Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering.** 43-48.

Snae, C. and Brueckner, M. (2009). Novel Phonetic Name Matching Algorithm with a Statistical Ontology for Analysing Names Given in Accordance with Thai Astrology. Informing Science and Information Technology. Vol.6. 497-515

Snae, C., Brückner, M. and Wongthongtham, P. 2007. Local Organization and Business Ontology (LOBO). **IEEE International Conference on Digital Ecosystems and Technologies.** 292-295.

Sylejmani, K. and Dika, A. 2011. Solving touristic trip planning problem by using taboo search approach. **IJCSI International Journal of Computer Science Issues.** Vol. 8. No.3. 139-149.

Winchester, I. 1970. The Linkage of Historical Records by Man and Computer: Techniques and Problems. **Journal of Interdisciplinary History**. Vol 1. 107-124.