# ทฤษฏีที่เกี่ยวข้อง

ทฤษฏี กระบวนการและ เครื่องมือต่างๆ ที่เกี่ยวข้องในการจัดทำโปรแกรมแสดงผลข้อมูลชนิดกราฟ ออกมาในรูปแบบกราฟิก มีดังนี้

2.1 กราฟและขั้นตอนวิธีของกราฟ

2.2 Python

2.3 Qt

2.4 GraphML

ซึ่งจะอธิบายถึงรายละเอียด วิธีใช้ ตัวอย่าง และการนำมาใช้ของแต่ล่ะหัวข้อข้างต้นโดยละเอียดต่อไปนี้

## กราฟ และ ขั้นตอนวิธีของกราฟ

ในทางคณิตศาสตร์ และวิทยาการคอมพิวเตอร์ กราฟ คือวัตถุพื้นฐานของการศึกษาในทฤษฎีกราฟ กล่าวอย่างไม่เป็นทางการได้ว่า, กราฟคือเซตของวัตถุที่เรียกว่า จุดยอด, จุดตัด (vertex, node) ซึ่งเชื่อมต่อกันด้วย เส้นเชื่อม (edge) ในบางการประยุกต์ใช้งาน เส้นเชื่อมอาจแสดงอย่างมีทิศทางได้

**นิยาม**

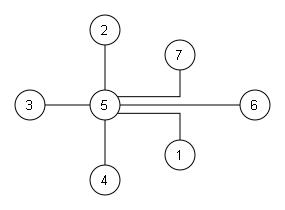
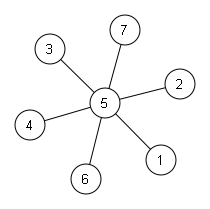
กราฟที่ไม่มีทิศทาง G = (V,E) คือโครงสร้างที่ประกอบด้วยกลุ่มของจุด V = {v1,v2,&,vm} และกลุ่มของเส้นเชื่อมระหว่างจุด 2 จุด E = {e1,e2, &, en} แต่ละเส้นเชื่อมจะต้องเกิดกับจุด 2 จุดที่ไม่เรียงลำดับ {m,v} จุด m และ v ไม่เป็นจุดเดียวกัน แต่ว่า {v,m} ถือว่าเป็นเส้นเดียวกันกับ {m,v}

graph

**รูปที่ 2-1** อธิบายความหมายกราฟ

ถ้ามีข้อมูลเป็นเซตของความสัมพันธ์ระหว่างเส้นเชื่อม และจุดตัด ชุดหนึ่งๆก็สามารถที่จะนำมาเขียนเป็นกราฟได้หลายรูปแบบในการแสดงผลแม้ว่า จะเป็นข้อมูลชุดเดียวกันก็ตาม

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |

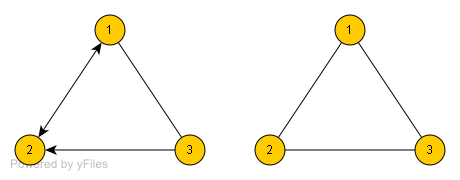


Organic graph layout Orthogonal graph layout adjacency matrix

**รูปที่ 2-2** ความแตกต่างของการแสดงผลกราฟ

**นิยาม**

กราฟที่มีทิศทาง G = (V, E) คือกราฟที่ประกอบด้วยกลุ่มของจุด V และกลุ่มของเส้นเชื่อม E โดยที่แต่ละเส้นเชื่อม จะเชื่อมจุด 2 จุด {u, v} ที่มีลำดับแน่นอน จากจุด u ไป v เส้นโยงดังกล่าวจะมีหัวลูกศรบอกทิศทาง จาก u ไป v โดยที่ u กับ v อาจจะเป็นจุดเดียวกันก็เป็นได้ หรือว่าจะมีการเชื่อมโยงกลับของv ไป u ก็เป็นได้

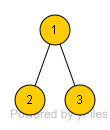
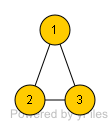


กราฟแบบมิทิศทาง กราฟแบบไม่มีทิศทาง

**รูปที่ 2-3** ความแตกต่างของกราฟแบบมีทิศทางและไม่มีทิศทาง

จากรูป 2-3 กราฟที่ เส้นเชื่อม มีทิศทางแม้เพียงเส้นเดียวก็ถือว่ากราฟนั้นเป็น กราฟที่มีทิศทางแล้ว ในที่นี้คือรูปตัวอย่างทางด้านซ้าย ส่วนทางด้านขวาคือกราฟไม่มีทิศทาง ซึ่งเรื่องของทิศทางนี้จะนำไปใช้พิจารณาในการจัดรูปแบบกราฟบางประเภท

กราฟยังแบ่งประเภทได้ด้วยการเชื่อมโยงของ เส้นเชื่อมได้อีกแบบคือ แบบต้นไม้(Tree) กับ แบบเครือข่าย (Network) โดย ต้นไม้คือกราฟที่ไม่เกิดการเชื่อมโยงเป็นวงขึ้นภายในตัวมันเองเลยและ แบบเครือข่ายคือกราฟที่มีการเชื่อมโยงกันเป็นวง ขึ้นภายในกราฟนั้นตั้งแต่ 1 วงขึ้นไป

ต้นไม้ เครือข่าย

**รูปที่ 2-4** ความแตกต่างของกราฟแบบต้นไม้และแบบเครือข่าย

### ขั้นตอนวิธีการหาต้นไม้แบบแผ่ที่เล็กที่สุด

**นิยาม**

ต้นไม้แบบแผ่ T ของกราฟ G = (V, E) คือต้นไม้ที่ประกอบด้วยจุดทุกจุดใน V และต้นไม้แบบแผ่ที่เล็กที่สุดคือต้นไม้แบบแผ่ที่มีผลรวมของน้ำหนักบนเส้นเชื่อมทั้งหมดน้อยที่สุด

**ขั้นตอนวิธีของ Prim**

เป็นวิธีในการหาต้นไม้แบบเล็กที่สุด โดยการสร้างต้นไม้แบบแผ่จากต้นเล็กๆ ก่อนแล้วค่อยๆ ขยายขนาดของต้นไม้จนกระทั่งมันคลุมจุดทุกจุดใน V ของกราฟ G = (V, E, W) วิธีของ Prim เป็นดังต่อไปนี้ ให้ VS เป็นที่เก็บกลุ่มของจุดสำหรับต้นไม้แบบแผ่ที่เล็กที่สุด และ T เป็นที่เก็บกลุ่มของเส้นเชื่อมของต้นไม้แบบแผ่ที่เล็กที่สุด โดยสามารถสรุปการทำงานได้เป็นขั้นตอนดังนี้

1. T = ∅; VS = ∅
2. เลือกจุดแรกตามใจชอบ สมมติ ว่าเลือก a;
3. V = V – {a}: VS = VS  ∪ {a};
4. while V ≠ ∅ do
   1. begin
      1. เลือกจุดที่อยู่ใน V ที่อยู่ติดกับจุดที่อยู่ใน VS โดยที่เส้นโยงระหว่างจุดที่เลือกกับจุดใน VS มีน้ำหนักน้อยที่สุด สมมติว่าจุดที่เลือกคือจุด U
      2. VS = VS ∪ {w}; T = TU {{b,w}} เมื่อ b คือจุดที่อยู่ติดกับ w   
         และ b ∈ VS; V = V – {w};
   2. end
5. end

ระยะเวลาที่ใช้ในการหาต้นไม้แบบแผ่โดยวิธีของ Prim เท่ากับ O(n2) เมื่อ n = ||V|| และ e = ||E||

### ขั้นตอนวิธีการหาระยะทางที่สั้นที่สุดระหว่างจุด 2 จุดในกราฟ

กราฟที่ใช้กับขั้นตอนวิธีนี้เป็นกราฟที่ไม่มีทิศทาง แต่ว่าเป็นกราฟที่มีน้ำหนักซึ่งน้ำหนักนี้ ใช้แทนระยะทางระหว่างจุด 2 จุด บนเส้นเชื่อมใดๆ ปัญหาของการหาระยะทางที่สั้นที่สั้นที่สุดระหว่างจุด 2 จุด พิจารณาได้โดยการสมมติการเดินระหว่างจุด a ไปจุด b โดยที่มีจุด x เชื่อมระหว่างเส้นทางจาก a ไป b และมีเส้นเชื่อม a ไป b โดยตรงด้วย ซึ่งจะพิจารณาว่าการระยะทาง a ไป x เป็น l(x) และ a ไป b เป็น l(b) และ w(x,b) คือระยะระหว่าง x และ b โดยกำหนดให้ x เป็นจุดหนึ่งใน Q ซึ่งอาจทำให้ l(x)+w(x,b) น้อยกว่า l(b) ดังนั้นวิธีการหาระยะทางที่สั้นที่สุดเป็นดังต่อไปนี้

กำหนดให้ V เป็นจุดเริ่มต้นทั้งหมด

1. เริ่มต้นให้ P = {a} และ Q = V – {a} สำหรับทุก ๆ จุด b ใน Q ให้ l(b) = w(a,b)
2. ให้ x เป็นจุดหนึ่งใน Q ซึ่งให้ค่า w(a,x) ต่ำสุดในบรรดาจุดทั้งหมดใน Q
3. ถ้าจุด x เป็นจุดที่ต้องการจุดสุดท้าย ให้หยุดการค้นหาระยะทาง ถ้าไม่ใช่ให้ P’ = P ∪ {x} และ Q’ = Q – {x} สำหรับทุกจุด b ให้ Q’ คำนวณหาระยะทางจากจุด x ไปยังจุด b ดังต่อไปนี้
4. l’(b) = min {l(b), l(x) + w(x ,b)}
5. ทำซ้ำในขั้นตอนที่ 2 โดยการแทน P ด้วย P’ และ Q ด้วย Q’ ค่าของ w(a, x) จะเท่ากับ **∞** ถ้าไม่มีเส้นเชื่อมระหว่างจุด a และจุด x

ระยะเวลาในการหาเส้นทางที่สุดในกราฟเท่ากับ O(n2) เมื่อ n = ||V||

## Python

* **ข้อมูลเบื้องต้น**

ไพธอน (Python) เป็นภาษาโปรแกรมในลักษณะภาษาอินเตอร์พรีเตอร์โปรแกรมมิ่ง (Interpreted programming language) ผู้คิดค้นคือ Guido van Rossum ในปี 1990 ซึ่ง Python เป็น การจัดการชนิดของตัวแปรแบบแปรผันตามข้อมูลที่บรรจุอยู่ (Fully dynamically typed) และใช้การจัดการหน่วยความจำเป็นอัตโนมัติ (Automatic memory management) โดยได้เป็นการพัฒนาและผสมผสานของภาษาอื่น ๆ ได้แก่ ABC, Modula-3, Icon, ANSI C, Perl, Lisp, Smalltalk และ Tcl และภาษา Python ยังเป็นแนวคิดที่ทำให้เกิดภาษาใหม่ ๆ ซึ่งได้แก่ Ruby และ Boo เป็นต้น

Python นั้นพัฒนาเป็นโครงการ Open source โดยมีการจัดการแบบไม่หวังผลกำไรโดย Python Software Foundation และสามารถหาข้อมูลและตัวแปรภาษาได้จากเว็บไซต์ของ Python เอง ที่ http://www.python.org/ ซึ่งในปัจจุบัน (ณ.วันที่ 29 กันยายน 2551) Python ได้พัฒนาถึงรุ่นที่ 2.5.2 (ออกวันที่ 22 กุมภาพันธ์ 2551) และรุ่นทดสอบการทำงาน หรือ beta นั้นอยู่ที่รุ่น 2.6rc2(12 กันยายน 2551) และ 3.0rc1 (17 กันยายน 2551)

* **หลักปรัชญาของภาษาไพธอน**

Python เป็นภาษาที่สามารถสร้างงานได้หลากหลายกระบวนทัศน์ (Multi-paradigm language) โดยจะมองอะไรที่มากกว่าการ coding เพื่อนำมาใช้งานตามรูปแบบเดิม ๆ แต่จะเป็นการนำเอาหลักการของกระบวนทัศน์ (Paradigm) แบบ Object-oriented programming, Structured programming, Functional programming และ Aspect-oriented programming นำเอามาใช้ทั้งแบบเดี่ยว ๆ และนำมาใช้ร่วมกัน ซึ่ง Python นั้นเป็น ภาษาที่มีการตรวจสอบชนิดตัวแปรแบบยืดหยุ่น (dynamically type-checked) และใช้ Garbage collection ในการจัดการหน่วยความจำ

* **ข้อเด่นของภาษาไพธอน**
* ง่ายต่อการเรียนรู้ โดยภาษา Python มีโครงสร้างของภาษาไม่ซับซ้อนเข้าใจง่าย ซึ่งโครงสร้างภาษา Python จะคล้ายกับภาษาซีมาก เพราะภาษา Python สร้างขึ้นมาโดยใช้ภาษาซี ทำให้ผู้ที่คุ้นเคยภาษาซี อยู่แล้วใช้งานภาษา Python ได้ไม่ยาก นอกจากนี้โดยตัวภาษาเองมีความยืดหยุ่นสูงทำให้การจัดการกับงานด้านข้อความ และ Text File ได้เป็นอย่างดี
* ภาษา Python ถูกสร้างขึ้นโดยได้รวบรวมเอาส่วนดีของภาษาต่างๆ เข้ามาไว้ด้วยกัน อาทิเช่น ภาษา ABC, Modula-3, Icon, ANSI C, Perl, Lisp, Smalltalk และ Tcl
* **Desktop GUIs**

เมื่อ Python ได้ติดตั้งลงบนเครื่องของคุณแล้ว จะมี Tk GUI development library ซึ่งเป็น libraries ที่มีความสามารถเทียบเท่า Microsoft Foundation Classes (MFC, ซึ่งคล้าย ๆ กับ win32 extensions), wxWidgets, GTK, Qt, Delphi และอื่น ๆ ทำให้สามารถพัฒนาซอฟต์แวร์ประยุกต์ต่าง ๆ แบบ Graphic user interface ได้

* **ซอฟต์แวร์ที่เขียนด้วยไพธอน**
* **BitTorrent**เป็นการพัฒนาโดยระบบการจัดการไฟล์ BitTorrent, การจัดการ การกระจายตัวของ Package ข้อมูลใน Tracker และการเข้ารหัสส่วนข้อมูลต่าง ๆ
* **Plone**ระบบ Content Management System
* **Zope**ระบบพัฒนาซอฟต์แวร์บนอินเทอร์เน็ตแบบ web-application platform
* **Battlefield 2** เกมส์ First Person Shooter ที่ได้ใช้ Python ในการทำ Configuration scripts
* **Indian Ocean Tsunami Detector**ซอฟต์แวร์สำหรับมือถือเพื่อแจ้งเตือน Tsunami
* **EVE Online** เกมส์แบบ Multi Massive Online Role Playing Game ซึ่งเป็นเกมส์ที่ได้รับอันดับสูงมากบน MMORPG.com

## Qt

Qt (อ่านว่า cute) เป็น cross-platform framework ในการพัฒนาโปรแกรมที่ถูกพัฒนาขึ้นโดย Trolltech เพื่อใช้ในการพัฒนาโปรแกรมในรูปแบบกราฟิกหรือเป็นที่รู้จักกันในอีกชื่อว่า widget-toolkit แล้วก็ยังถูกใช้ในพัฒนาโปรแกรมที่เป็นแบบ Text mode หรือว่าโปรแกรมที่ทำงานอยู่บนเซิร์ฟเวอร์ก็ได้ Qt ถูกใช้อย่างเห็นได้ชัดในหลายโปรแกรมเช่น KDE(Linux Desktop Environment) ,Opera (Web browser),Google Earth, Skype, Photoshop Element ,VirtualBox ,VLC Media Playerเป็นต้น

Qt สร้างขึ้นมาโดยใช้ C++ หลายส่วนขยายของโปรแกรมที่ไม่เป็นมาตรฐานถูกเขียนขึ้นโดยเพิ่มเข้ามากับมาโค้ดที่เป็นมาตรฐานของ C++ ก่อนที่จะ compile ออกมาเป็น Qt โดยเป้าหมายการการพัฒนาโปรแกรมของ Qt มีแนวความคิดมากจากคำว่า

*“Code less Create more Deploy everywhere”*

*“เขียนโค้ดน้อยลง สร้างสรรค์ผลงานได้มากขึ้น นำไปใช้ได้ทุกที่”*

ดังนั้น Qt แม้ว่าจะเขียนขึ้นมาด้วย C++ แต่ว่าก็สามารถที่นำไปใช้กับภาษาโปรแกรมภาษาอื่นได้ เช่น Ada (QtAda), C# (Qyoto/Kimono),Java(Qt Jambi), Pascal, Perl, PHP(PHP-Qt),Ruby( RubyQt) และภาษาที่ผู้พัฒนาได้เลือกมาใช้กับ framework นี้คือ Python(PyQt) ซึ่งทั้งตัวภาษาและ framework สามารถทำงานได้บน platforms หลักๆ ได้ทั้งหมดและยังสนับสนุนเรื่องภาษาท้องถิ่นจำนวนมากรวมถึงภาษาไทย มีความสามารถอื่นๆที่ไม่ใช่ด้านกราฟิกเพื่อสนับสนุนการเขียนโปรแกรมอีกได้แก่ SQL, XML, การจัดการกับการทำงานแบบ Thread ,Network ,และการจัดการไฟล์ข้าม platform

### PyQt4

เซตของ python แปลงมาจาก Qt4 เพื่อให้สามารถนำมาเขียนได้ง่ายในรูปแบบภาษาแบบ python ซึ่งได้รวมเอาความสามารถต่างๆ ของ Qt4 ไว้ครบถ้วนซึ่งผู้ใช้ไม่ต้องเรียนรู้ Syntax ของภาษาใหม่แต่อย่างใด แต่ต้องเรียนรู้วิธีการใช้ และหลักการที่เราจะนำมาใช้ในการพัฒนาโปรแกรมซึ่งทำให้ลดเวลาในการปรับตัวลงได้มาก และทำให้เราสามารถสร้างงานได้มากขึ้น

## GraphML

GraphML เป็น xml ชนิดหนึ่งซึ่งออกแบบมาเพื่อการเก็บข้อมูลชนิดกราฟ ซึ่งออกแบบโดย Graphdrawing.org โดยการออกแบบนั้นครอบคลุมทุกเนื้อหา และง่ายที่จะนำมาใช้กับข้อมูลประเภทกราฟ ซึ่งมันประกอบไปด้วยส่วนประกอบหลักของตัวภาษาที่ใช้ในการอธิบายข้อมูลประเภทกราฟ และยืดหยุ่นพอสำหรับที่จะให้มีส่วนเสริม เพิ่มเข้าไปสำหรับข้อมูลเฉพาะของแต่ละโปรแกรมที่นำไปใช้ ซึ่งส่วนประกอบหลักของตัวภาษาจะประกอบไปด้วย

* กราฟแบบมีทิศทาง กราฟไม่มีทิศทาง กราฟแบบผสม
* Hypergraphs
* Hierarchical graphs
* Graphical representations
* References to external data
* Application-specific attribute data
* Light-weight parsers

ซึ่ง GraphML ได้ถูกนำไปใช้ในหลายๆ โปรแกรมที่ใช้ข้อมูลประเภทกราฟ และมีอีกหลายๆโปรแกรมที่ให้การสนับสนุนการใช้ GraphML เช่น JUNG, yFile, AT&T Research, IBM Research ซึ่งผู้พัฒนาได้นำเอาไฟล์ประเภทนี้มาใช้ในการเก็บข้อมูลประเภทกราฟ แต่ว่าได้นำความสามารถของ GraphML มาแค่บางส่วนเท่านั้น

### โครงสร้างของไฟล์ประเภท GraphML

โครงสร้างของ GraphML โดยทั่วไปจะเหมือนกับ xml แต่ว่า จะมีการกำหนดโครงสร้างของภาษาถูกกำหนดอยู่ใน GraphML Schema ซึ่งตัวแฟ้ม Schema นี้ถูกเก็บอยู่ที่ลิงค์นี้ http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd โดยใน Schema จะกำหนดข้อกำหนดพื้นฐานของ GraphML เช่น รากของ GraphML จะต้องเป็น <graphml> เสมอแล้วลูกขั้นต่อไปก็จะเป็น <graph> และมีการกำหนด จุดตัดและเส้นเชื่อมด้วย <node> และ <edge> ตามลำดับ ซึ่งการประมวลผลกับไฟล์ประเภทนี้ ไม่จำเป็นต้องอ่านข้อมูลทั้งไฟล์ขึ้นมา เช่น แท็กอย่าง <port>, <hyperedge>, <endpoint> ซึ่งอาจจะเป็นแท็กที่ไม่สนใจ ก็ไม่จำเป็นต้องอ่านมาประมวลผล และนี่คือตัวอย่างไฟล์ขนาดเล็กที่เก็บข้อมูลของกราฟไม่มีเส้นเชื่อม ที่มีจุดตัด 2 จุดและ 1 เส้นเชื่อม

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <graphml xmlns="http://graphml.graphdrawing.org/xmlns"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">  <graph id="G" edgedefault="undirected">  <node id="n0"/>  <node id="n1"/>  <edge id="e1" source="n0" target="n1"/>  </graph>  </graphml> |