เทคนิคการโปรแกรมภาษาซีสำหรับขั้นตอนวิธีเชิงพันธุกรรม ในงานคำนวณทางวิศวกรรมไฟฟ้า

C Language Programming Techniques for Genetic Algorithm in Electrical Engineering computations

ชัยณรงค์ วิเศษศักดิ์วิชัย[1] ประเสริฐ เผ่าชู[2] และ สายชล ชุดเจือจีน[1]

[1]สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลกรุงเทพ

[2]สาขาวิชาคณิตศาสตร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลกรุงเทพ

โทรศัพท์ : 08-1343-7693 E-mail: chainarong.w@rmutk.ac.th

บทคัดย่อ

โครงสร้างของการโปรแกรมภาษาซี มีความเหมาะสมกับงานคำนวณทางวิศวกรรม โดยเฉพาะกับเทคนิคการโปรแกรมเชิงวัตถุ(Object-Oriented Programming; OOP) ที่จะทำการนำเสนอ ได้ดำเนินการบนภาษาการโปรแกรมซีพลัสพลัส(C++) ซึ่งเป็นส่วนขยายของภาษาซี วัตถุทางโปรแกรมของภาษาได้ถูกใช้เพื่อการสร้างขั้นตอนวิธีเชิงพันธุกรรมโดยจัดเป็นนามธรรมทางข้อมูล สำหรับปัญหาทางคณิตศาสตร์ในงานคำนวณทางวิศวกรรมไฟฟ้า ตัวตั้งต้นแบบ(constructor) ของวัตถุบนฐานคลาสที่สร้างขึ้นได้ถูกใช้ในเป้าหมายการกำหนดค่าเริ่มต้นให้กับโครโมโซม(chromosome) ประกอบด้วยขนาด รูปแบบ และจำนวนยีน(gene) ของขั้นตอนวิธีเชิงพันธุกรรม ที่ซึ่งลักษณะสมบัติประจำตัวต่าง ๆ ของโครโมโซมจะถูกนิยามด้วยสมาชิกข้อมูลของคลาส C++ ขณะที่ตัวดำเนินการเชิงพันธุกรรมและกระบวนการวิวัฒนาการจะถูกดำเนินงานด้วยสมาชิกฟังก์ชันวัตถุเชิงพันธุกรรมของบทความนี้ ให้ผลเป็นที่น่าพอใจสำหรับผลเฉลยเชิงตัวเลขของปัญหาการจ่ายกำลังไฟฟ้าอย่างประหยัด และการไหลของภาระในระบบกำลังไฟฟ้า

คำสำคัญ: การโปรแกรมเชิงวัตถุ ขั้นตอนวิธีเชิงพันธุกรรม การจ่ายกำลังไฟฟ้าอย่างประหยัด การไหลของภาระ

Abstract

The structure of C language programming is an appropriate for engineering calculations. The C language programming, in particular for the OOP (Object-Oriented Programming), having present in this paper which is implemented by the extended C language, also known as the C++ language programming. The programmatic objects of language are used to realize the genetic algorithm that is the data abstraction of the OOP for mathematic problems in electrical engineering computations. The constructors of class based object are used to initializing purpose for the chromosome consisting of the size, form and number of genes in the genetic algorithm. Where the attributes of chromosome are defined by the data member of C++ class while the genetic operators and evolutionary process will be implemented by the function members. The genetic object of this article gives satisfactory results for the numerical solutions in the economic power dispatch and load flow problems of electrical power system.

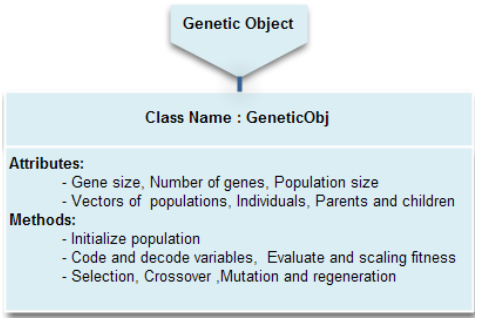
Keywords: Object-oriented programming, Genetic algorithm, Economic power dispatch, Load flow

บทนำ

การใช้คอมพิวเตอร์ สำหรับแก้ปัญหาทางคณิตศาสตร์ในงานวิศวกรรม ด้วยการพัฒนาโปรแกรมคอมพิวเตอร์ขึ้นจากขั้นตอนวิธีของระเบียบวิธีเชิงตัวเลขได้มีมาอย่างต่อเนื่องจนกระทั่งถึงปัจจุบัน ทั้งนี้ จะมีภาษาการโปรแกรม เป็นปัจจัยในการกำหนดหลักวิธีการโปรแกรม กรณีที่การคำนวณในระบบทางวิศวกรรมมีความยากและซับซ้อน หลักวิธีการโปรแกรมจะมีความสำคัญมาก ดังจะเห็นได้จากการนำเสนอวิธีใช้การโปรแกรมเชิงวัตถุ สำหรับการวิเคราะห์ระบบกำลังไฟฟ้า [1] โดยแสดงการสร้างตัวแบบที่มีลักษณะความเป็นนามธรรมของข้อมูล ประกอบด้วยโครงสร้างข้อมูลและวิธีดำเนินการให้กับองค์ประกอบของระบบกำลังไฟฟ้า อย่างไรก็ตาม เนื่องจากความไม่เป็นเชิงเส้นของระบบสมการกำลัง ไฟฟ้า ทำให้การหาผลเฉลยเชิงตัวเลขของปัญหาทางคณิตศาสตร์ที่เกิดขึ้นมีความยาก และต้องใช้ระเบียบวิธีเชิงตัวเลขที่เหมาะสม ดังปัญหาการจ่ายกำลังไฟฟ้าอย่างประหยัด [2] ได้มีความพยายามที่จะหาผลเฉลยเชิงตัวเลขด้วยการใช้ขั้นตอนวิธีเชิงพันธุกรรม แทนการใช้ระเบียบวิธีเชิงตัวเลข สำหรับการค้นหาจุดต่ำสุดของฟังก์ชันเป้าหมาย(objective function) ซึ่งก็คือราคาสุทธิของการผลิตกำลังไฟฟ้าของระบบกำลังไฟฟ้า ภายใต้เงื่อนไขบังคับ(constraints) ขั้นตอนวิธีเชิงพันธุกรรมไม่ได้มีความสามารถเพียงการหาผลเฉลยให้กับปัญหาการหาค่าเหมาะที่สุดทางคณิตศาสตร์ (mathematic optimization)เท่านั้น แต่ยังสามารถใช้ในการหาผลเฉลยให้กับระบบสมการไม่เชิงเส้น [3] ดังนั้นบทความนี้จึงนำเสนอการพัฒนาวัตถุเชิงพันธุกรรมด้วยภาษาการโปรแกรมซีพลัสพลัส เพื่อให้สามารถใช้ภาษาการโปรแกรมในงานคำนวณทางวิศวกรรมไฟฟ้า สำหรับการหาผลเฉลยทางคณิตศาสตร์ที่เกิดขึ้น

หลักการที่เกี่ยวข้อง

คลาส(class) ของภาษาการโปรแกรมซีพลัส พลัสจะประกอบด้วยสมาชิก 2 ชนิดคือ สมาชิกข้อมูล (data members) และสมาชิกฟังก์ชัน (function members) วัตถุเชิงพันธุกรรมของบทความจึงเป็นผลจากการสร้างคลาส [4] โดยวางแบบคลาสที่กำหนดชื่อให้เป็น GeneticObj สำหรับการประกาศวัตถุทางภาษาการโปรแกรมซีพลัสพลัส และจัดให้มีองค์ประกอบแสดงดังรูปที่ 1



รูปที่ 1 องค์ประกอบของคลาสสำหรับวัตถุเชิงพันธุกรรม

คลาสของภาษาการโปรแกรมซีพลัสพลัสในทางภาษาการโปรแกรมซีพลัสพลัส สมาชิกทั้งหมดของคลาสจะบรรจุอยู่ในตัวแบบคลาสที่มีรูปทั่วไปดังรูปที่ 2

```
class  ClassName
   {  public:
            //public member

      private:
            //private member
   }
```

รูปที่ 2 รูปทั่วไปของตัวแบบคลาสที่ใช้สร้างวัตถุเชิงพันธุกรรม

การกำหนดค่าตั้งต้นให้กับสมาชิกข้อมูลของคลาสรวมทั้งระเบียบวิธี (Methods) ของสมาชิกฟังก์ชัน สามารถดำเนินการโดยใช้ตัวแบบสมาชิกฟังก์ชัน ที่มีรูปทั่วไปดังรูปที่ 3

```
TypeDefinition   ClassName ::
FunctionName(arg1, ...)
   {  Declarations
      Statements
      return ;
   }
```

รูปที่ 3 รูปทั่วไปของตัวแบบฟังก์ชันที่ใช้สร้างวัตถุเชิงพันธุกรรม

ตัวแปรและประชากร

ขั้นตอนวิธีเชิงพันธุกรรม มีธรรมชาติในการค้นหาค่าสูงสุด [5] ในกรณีของการหาค่าสูงสุดของฟังก์ชันที่ประกอบด้วย n ตัวแปร ถ้อยแถลงทางคณิตศาสตร์สามารถเขียนออกมาได้ดังต่อไปนี้

$$maximize \quad f(\mathbf{x}) : subject\ to \quad l_i \le x_i \le u_i; i = 1\ to\ n$$

ระเบียบวิธีการคำนวณของขั้นตอนวิธีเชิงพันธุกรรมจะจำลองกระบวนการวิวัฒนาการทางธรรมชาติ ด้วยการสร้างโครโมโซมที่ประกอบด้วยยีน(genes) [6] จำนวนเท่ากับ n ตัวแปรตามปัญหาทางคณิตศาสตร์ข้างต้น

$$G_i = \{0,1\}^m \ and \ \ C_i = G_1 G_2 G_3 \quad G_n$$

เมื่อยีน G_i คือสายอักขระเลขฐานสอง m บิตและโครโมโซม C_i ประกอบด้วย n ยีน ด้วยเหตุนี้จึง สามารถสร้างประชากร (populations) ของขั้นตอนวิธีเชิงพันธุกรรมได้ในลักษณะ

$$P^t = \left\{C_1^t, C_2^t, \quad , C_S^t\right\}$$

เมื่อประชากร P^t รุ่นในรอบที่ t ประกอบด้วย s โครโมโซม สำหรับการคำนวณหาผลเฉลย

การเข้ารหัสและถอดรหัสตัวแปร

เมื่อตัวแปรของปัญหาทางคณิตศาสตร์ $l_i \le x_i \le u_i$ ถูกจำลองค่าด้วยยีน $G_i = \{0,1\}^m$ ซึ่งหมายความว่าค่าของผลเฉลย จะถูกแสดงแทนด้วยค่าดีสครีต (discrete representations) ที่มีทั้งหมด $N = 2^m - 1$ ช่วงกว้างและ

$$x_i = l_i + \frac{u_i - l_i}{2^m - 1}\left(b_0 2^0 + b_1 2^1 + b_2 2^2 + \quad + b_{m-1} 2^{m-1}\right)$$

เมื่อ $b_i = \{0,1\}$ คือบิตที่ i ของ G_i ที่มี b_0 และ b_{m-1} เป็นบิตนัยสำคัญน้อยสุด และบิตนัยสำคัญสูงสุดตามลำดับ

ค่าความเข้มแข็ง(Fitness Values)

ประชากร $P = \left\{C_1, C_2, \quad , C_S\right\}$ ที่ประกอบด้วย s โคร โมโซมจะมีค่าความเข้มแข็ง $f_i = f[C_i]$ โดย

$$f[C_i] = f(G_1, G_2, G_3, \quad , G_n)$$

ค่าความเข้มแข็ง f_i นี้จะถูกประเมินทุกรอบของการวิวัฒนาการด้วยขั้นตอนวิธีเชิงพันธุกรรม [7]

บ่อผสมพันธุ์(Mating Pool)

ในบ่อผสมพันธุ์สมาชิกที่อ่อนแอจะถูกแทนที่ด้วยสมาชิก ที่เข้มแข็ง ซึ่งขึ้นอยู่กับค่าความเข้มแข็ง โดยบทความนี้ใช้การคัดเลือกวงล้อรูเล็ต(roulette wheel selection) ถ้า f_i คือค่าของฟังก์ชันที่ประเมินจากโครโมโซม C_i ของประชากร P การคัดเลือกจะทำการปรับขนาด f_i ด้วย [8]

$$f_i' = \frac{f_i + C}{D}$$

เมื่อ C = 0.1 f_h – 1.1 f_l และ $D = \max(1, f_h + C)$ โดย f_h และ f_l เป็นค่าของฟังก์ชันที่ประเมินได้มากสุด และน้อยสุดตามลำดับ

โครโมโซม C_j จะถูกเลือกจากความจริงของอสมการต่อไปนี้

$$f_1' + f_2' + \quad + f'_{j-1} \le rS \le f_1' + f_2' + \quad + f'_j$$

เมื่อ $S = \sum_{i=1}^{s} f_i'$ และ r คือค่าสุ่มในย่าน 0 ถึง 1

การสืบพันธุ์(reproduction)

ประชากรรุ่นใหม่จะถูกสร้างขึ้นจากบ่อผสมพันธุ์ ในที่นี้ได้ใช้วิธีดำเนินการสลับสายพันธุ์เชิงเดี่ยว(simple cross over operation) ด้วยการใช้สองโครโมโซมพ่อแม่พันธุ์ และทำการสุ่มจำนวนเต็ม k ในย่านจำนวนเต็ม $1 \le k \quad \le nm-1$ ดังนั้นถ้ากำหนดให้ C_i และ C_j คือโครโมโซมสองพ่อแม่พันธุ์

$$C_i = \left[b_m b_{m-1} \quad b_k b_{k-1} \quad b_3 b_2 b_1\right]$$
$$C_j = \left[b'_m b'_{m-1} \quad b'_k b'_{k-1} \quad b'_3 b'_2 b'_1\right]$$

ของประชากร P ที่มีประชากรทั้งหมด s โครโมโซม และเขียนแทนวิธีดำเนินการสลับสายพันธุ์เชิงเดี่ยวด้วยตัวดำเนินการ“ $\oplus$ ”ทำให้สองโครโมโซมรุ่นใหม่สามารถเขียนออกมาได้ในลักษณะ

$$C_i^{+j} = \left(C_i \oplus C_j\right) \text{ และ } C_j^{+i} = \left(C_j \oplus C_i\right)$$

โดยที่ C_i^{+j} และ C_j^{+i} เป็นคู่สลับสายพันธุ์เชิงเดี่ยวที่มีอันดับ (ordered crossover pair) การสลับสายพันธุ์เชิงเดี่ยวจะมีผลดำเนินการดังนี้

$$C_i^{+j} = \left[b_m b_{m-1} \quad b'_k b'_{k-1} \quad b'_3 b'_2 b'_1\right]$$
$$C_j^{+i} = \left[b'_m b'_{m-1} \quad b_k b_{k-1} \quad b_3 b_2 b_1\right]$$

การกลายพันธุ์(mutation)

เนื่องจากมีโอกาสที่การคัดเลือกพ่อแม่พันธุ์ ด้วยการคัด เลือกวงล้อรูเล็ต และการสืบพันธุ์ด้วยวิธีดำเนินการสลับสายพันธุ์เชิงเดี่ยวจะทำให้ได้ประชากรลูกหลานที่เหมือนกันทุกประการและไม่เข้มแข็ง การเปลี่ยนแปลงโอกาสให้ได้ประชากรมีค่าความเข้มแข็งที่คาดหวังสามารถทำได้ด้วยวิธีดำเนินการแบบสุ่มเพื่อทำการกลายพันธุ์ วิธีการนี้ทุกบิตของทุกโครโมโซมของประชากร จะถูกพิจารณาด้วยการสุ่มจำนวนค่าใด ๆ r ที่อยู่ในย่าน $0 \le r \le 1$ ถ้าพบว่าในแต่ละบิตที่พิจารณานั้นและทำการสุ่มจำนวนขึ้น มาได้ค่า $r < b_p$ เมื่อ b_p เป็นค่ากำหนด บิตนั้นจะถูกคอมพลีเมนต์ (complement)

วิธีการดำเนินงาน

การวางแบบวัตถุเชิงพันธุกรรม สามารถเริ่ม ต้นด้วยการพิจารณาประชากร $P = \left\{C_1, C_2, \quad, C_s\right\}$ สำหรับ $C_i = G_1 G_2 G_3 \quad G_n$ $and \ G_i = \{0,1\}^m$ เห็นได้ชัดว่าจำนวนเต็ม m, n และ s จะเป็นข้อมูลพื้นฐานที่สุดของประชากร P

การสร้างสมาชิกข้อมูลของคลาสสำหรับวัตถุเชิงพันธุกรรม

ทำการใช้ข้อมูลพื้นฐานที่สุดของประชากร P กำหนดตัวแปรหน่วยความจำประเภทจำนวนเต็ม ให้กับจำนวนเต็ม m, n และ s ดังจะเห็นได้ในรูปที่ 4 ภายในคลาส GeneticObj ได้ทำการประกาศ GeneSize, GeneNo และ PopuSize สำหรับจำนวนเต็มข้างต้นตามลำดับ นอกจากนี้ค่าความเข้มแข็ง $f_i = f[C_i]$ และการปรับขนาด f_i ด้วย $f_i' = \frac{f_i + C}{D}$ ภาย ในคลาสได้ใช้ตัวแปรหน่วยความจำ Fitness และ ScaledFitness เพื่อจัดเก็บผลการคำนวณทั้ง 2 ค่าตามลำดับด้วย

```
class GenaticObj {
   public:
      int GeneSize, GeneNo, PoppuSize, VariableNo;
      int *SelectionIndex;
      long double *Fitness, *ScaledFitness, *Solution;
      std::vector<double> *VarIntervals;
      std::vector<double> *PopuVariables;
      std::vector<bool> *Individuals;
      std::vector<bool> *ParentIndividuals;
      std::vector<bool> *ChildIndividuals;
      GenaticObj(void) { }
      GenaticObj(int, int, int, int, double [ ][2]);
      void DecodingVariables(void);
      void FitnessValue(void);
      void FitnessScaling(void);
      long double ObjectiveFunc(long double *);
      void Selection(void);
      void CrossOver1P(void);
      void Mutation(void);
      void ReGeneration();
      private:
   };
```

รูปที่ 4 สมาชิกภายในวัตถุเชิงพันธุกรรม GeneticObj

ในการจัดเก็บสมาชิก C_i ของประชากร P บทความนี้ได้ทดลองใช้วัตถุเชิงเวกเตอร์ (vector object) ของ ภาษาการโปรแกรมซีพลัสพลัส ทำให้ค่าทั้งหมดของถ้อยแถลงทางคณิตศาสตร์ต่อไปนี้

$$C_i = G_1 G_2 G_3 \quad G_n \quad ; G_i = \{0,1\}^m$$

สามารถทำการจัดเก็บลงในตัวแปรหน่วยความจำชื่อ Individuals ด้วยข้อความสั่งต่อไปนี้ ภายในคลาส GeneticObj

```
std::vector<bool>Individuals;
```

การคัดเลือกสมาชิกของประชากร P สำหรับพ่อแม่พันธุ์และทำการผสมพันธุ์ให้ได้ประชากร จะต้องใช้วัตถุเชิงเวกเตอร์สำหรับจัดเก็บข้อมูลลงในตัวแปรหน่วยความจำเพิ่มเติมคือ

```
std::vector<bool>
      *ParentIndividuals;
std::vector<bool>
      *ChildIndividuals;
```

การสร้างสมาชิกฟังก์ชันของคลาสสำหรับวัตถุเชิงพันธุกรรม

ปกติในการนำวัตถุทางภาษาการโปรแกรมไปดำเนินงาน จะต้องมีการตั้งค่าเริ่มต้นให้กับวัตถุนั้นด้วยตัวตั้งต้นแบบ ซึ่งจัดเป็นสมาชิกฟังก์ชันของคลาส ในการนี้ได้สร้างตัวตั้งต้นแบบ GeneticObj สำหรับกำหนดประชากรเริ่มต้น $P^0 = \left\{C_1^0, C_2^0, \quad , C_s^0\right\}$ โดยผ่านอาร์กิว เมนต์ m, n และ s ให้กับตัวตั้งต้นแบบ และทำการกำหนด $C_i^0 = \left[b_m b_{m-1} \quad b_k b_{k-1} \quad b_3 b_2 b_1\right]$ ด้วยการสุ่มเทียม(pseudo random) $0 \le r_k \le 1$;$1 \le k \le m$ ดังนี้

$$b_k = \begin{cases} 0; & r_k \le 0.5 \\ 1; & r_k > 0.5 \end{cases}$$

ซึ่งเขียนเป็นข้อความสั่งภาษาการโปรแกรมซีพลัสพลัสได้คือ

```
Bk = (double) rand()/RAND_MAX
   <= 0.5 ? 0 : 1;
```

เมื่อ Bk คือตัวแปรหน่วยความจำ rand() คือฟังก์ชันสุ่มเทียมซีพลัสพลัส และ RAND_MAX คือค่าคงที่สูงสุดของการสุ่มทั้งนี้ทุกสมชิก C_i ของ P ถูกวางแบบให้มีวัตถุเชิงเวกเตอร์ต่อไปนี้

```
std::vector< double>
      *PopuVariables;
```

เป็นคู่ตอบสนองค่าโดยการถอดรหัส ดังนั้นด้วยการประกาศวัตถุเชิงพันธุกรรมของคลาส GeneticObj ดังต่อไปนี้

```
GeneticObj GAobj_1;
```

ทำให้หลังจากตั้งค่าเริ่มต้นให้หน่วยวัตถุ (instance) เชิงพันธุกรรม GAobj_1 การถอดรหัสให้ C_i เพื่อกำหนดค่าลงใน PopuVariables จะทำได้ด้วยข้อความสั่ง

```
GAobj_1.DecodingVariables();
```

เมื่อ DecodingVariables เป็นสมาชิกฟังก์ชันของคลาสตามรูปที่ 4

ระเบียบวิธีของคลาสขั้นตอนวิธีเชิงพันธุกรรม

ถ้ากำหนดให้ $\bar{C}_i$ คือค่าถอดรหัสของ C_i ระเบียบวิธีของคลาส GeneticObj จะประเมินค่าโดยใช้ $f_i = f[\bar{C}_i]$ เพื่อคำนวณค่าความเข้มแข็ง ในกรณีของปัญหาการคำนวณค่าสูงสุดจะคำนวณค่าความาเข้มแข็ง จาก f_i' ตามที่แสดงก่อนหน้า แต่สำหรับกรณีปัญหาการคำนวณค่าต่ำสุดจะคำนวณค่าความเข้มแข็งจาก

$$f_i' = \frac{D}{f_i + C}$$

ดังนั้น ด้วยหน่วยวัตถุเชิงพันธุกรรมที่สร้างขึ้นค่าความเข้มแข็งจะคำนวณด้วยข้อความสั่งต่อไปนี้

GAobj_1.FitnessValue();

GAobj_1.ScaledFitnessValue();

ทั้งฟังก์ชัน FitnessValue และ ScaledFitness Value จัดเป็นระเบียบวิธีของคลาส GeneticObj เมื่อทำการคำนวณค่าความเข้มเข็งให้กับ $P^0 = \left\{C_1^0, C_2^0, \quad, C_s^0\right\}$ สำหรับหน่วยวัตถุเชิงพันธุกรรมแล้ว การคัดเลือกประชากรของพ่อแม่พันธุ์ จะใช้ความจริงของอสมการ

$$f_1' + f_2' + \quad + f'_{j-1} \le rS \le f_1' + f_2' + \quad + f'_j$$

ทั้งนี้ค่าสุ่ม r คือค่าสุ่มในย่าน 0 ถึง 1 ที่ยังคงใช้ข้อความสั่งการสุ่มเทียมของภาษาการโปรแกรมซีพลัสพลัส ที่มีลักษณะดังนี้

r =(double)rand()/RAND_MAX;

เมื่อ r คือตัวแปรหน่วยความจำ ค่า j ที่ทำให้อสมการข้างต้นเป็นความจริง จะเป็นดัชนีเพื่อใช้ระบุสมาชิกของประชากร C_j ที่ถูกคัดเลือกให้เป็นพ่อแม่พันธุ์ โดยจะทำการคัดเลือกให้ได้ตามจำนวน s ครั้ง เท่ากับจำนวนประชากร P การคัดเลือกจะใช้ฟังก์ชันสมาชิก Selection ของคลาสในรูปที่ 4

GAobj_1.Selection();

ผลของระเบียบวิธี Selection จะทำให้ได้ค่าดัชนี j ระบุสมาชิกของประชากร C_j คัดไว้สำหรับประชากรพ่อแม่พันธุ์จำนวน s ค่า โดยคลาส GeneticObj ที่สร้างขึ้นนี้ได้ใช้ข้อความสั่ง

int *SelectionIndex;

ในการกำหนดตัวแปรหน่วยความจำใช้จัดเก็บดัชนีประชาระชากรของพ่อแม่พันธุ์ และการคำนวณประชากรรุ่นใหม่จะใช้ประชากรพ่อแม่พันธุ์เพื่อดำเนิน การสลับสายพันธุ์เชิงเดี่ยวด้วยฟังก์ชันสมาชิก Cross Over1P ที่สร้างขึ้นเพื่อการคำนวณประชากรรุ่นใหม่ C_i^{+j} และ C_j^{+i} แทนประชากรพ่อแม่พันธุ์ C_i และ C_j ทำให้ประชากรรุ่นใหม่ยังคงมีจำนวนประชากร s เท่าเดิม โดยประชากรรุ่นใหม่ทั้งหมด จะถูกกลายพันธุ์ด้วยฟังก์ชันสมาชิก Mutation ข้อความสั่งที่ใช้ในการกลายพันธุ์จะมีลักษณะดังนี้

ChildIndividuals[i][j]=
(double)rand()/RAND_MAX < Bp ?
ChildIndividuals[i][j].flip():
ChildIndividuals[i][j];

เมื่อ $1 \le i \le s$ และ $1 \le j \le nm$ และตัวแปรหน่วยความจำ Bp คือค่าคงที่กำหนดสำหรับการคอมพลีเมนต์ในทุกบิตของประชากรรุ่นใหม่ ฟังก์ชัน flip() ของวัตถุเชิงเวกเตอร์ในภาษาการโปรแกรมซีพลัสพลัส จะทำหน้าที่ในการคอมพลีเมนต์บิต ดังนั้นการสืบพันธุ์ของวัตถุเชิงพันธุกรรมที่สร้างขึ้นนี้สามารถทำได้โดยใช้ข้อความสั่ง

GAobj_1.CrossOver1P();

GAobj_1.Mutation();

ประชากรรุ่นลูกจะถูกนับเป็นประชากรรุ่นใหม่ดังนี้

$$P^t = \left\{C_1^t, C_2^t, \quad, C_s^t\right\}$$

เมื่อ $t = 1, 2, 3, \ldots, \mathrm{z}$ โดยค่า z จะเป็นเงื่อนไขในการหยุดคำนวณ การจัดตั้งประชากรรุ่นใหม่จะใช้สมาชิกฟังก์ชันดังนี้

GAobj_1.ReGeneration

ผลการดำเนินงาน

ด้วยวัตถุเชิงพันธุกรรมที่สร้างขึ้น สามารถใช้ในการหาผลเฉลย ให้กับปัญหาการจ่ายกำลังไฟฟ้าอย่างประหยัด [9] ดังต่อไปนี้

$$\mathcal{L} = F_t + \lambda(P_D + P_L - \sum_{i=1}^{ng} P_i) + \sum_{i=1}^{ng} \mu_{i(\max)} (P_i - P_{i(\max)}) + \sum_{i=1}^{ng} \mu_{i(\min)} (P_i - P_{i(\min)})$$

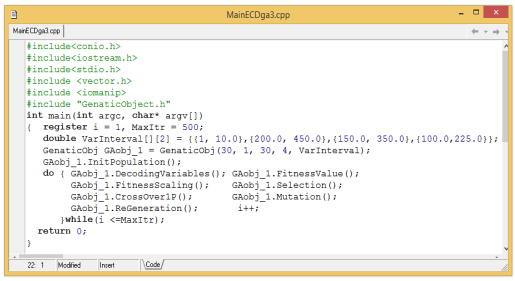
เมื่อ F_t, P_D, P_L และ P_i คือตันทุนเชื้อเพลิงสุทธิ ความต้องการกำลังสุทธิ ความสูญเสียในการส่งจ่ายกำลังไฟฟ้า และกำลังจ่ายไฟฟ้าหน่วยที่ i ตามลำดับ ขณะที่ L คือฟังก์ชันเป้าหมายระเบียบวิธีตัวคูณลากรองจ์ (Lagrange multiplier) โปรแกรมภาษาซีในรูปที่ 5 (ตัดคำสั่งพิมพ์ผล) แสดงการใช้ GeneticObj วัตถุเชิงพันธุกรรมของบทความ โดยใช้ข้อมูลการทดลองดังนี้

$$F_1 = 500 + 5.3P_1 + 0.004P_1^2 \quad ;200 \le P_1 \le 450$$

$$F_2 = 400 + 5.5P_2 + 0.006P_2^2 \quad ;150 \le P_2 \le 350$$

$$F_3 = 200 + 5.8P_3 + 0.009P_3^2 \quad ;100 \le P_3 \le 225$$

ความต้องการกำลังไฟฟ้า 975 MW ไม่คิดกำลังสูญเสียของการส่ง

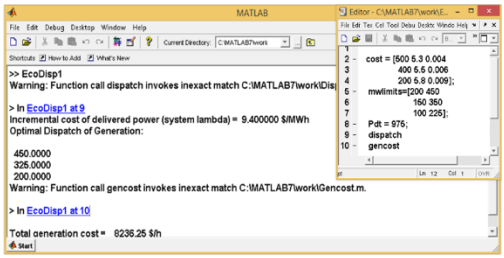


รูปที่ 5 การใช้วัตถุเชิงพันธุกรรมกับปัญหาการจ่ายกำลัง ไฟฟ้า

ผลการทดลองแสดงไว้ในรูปที่ 6 โดย P_O ทางด้านซ้าย แสดงประชากรเริ่มต้นของวัตถุเชิงพันธุกรรม GeneticObj ส่วน P_t ทางด้านขวาเป็นประชากรรุ่นรอบที่ 500 ที่แสดงออกถึงการลู่เข้า

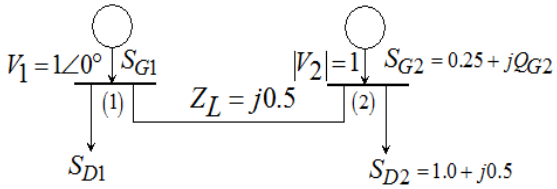
รูปที่ 6 ผลการใช้ GeneticObj กับปัญหาการจ่ายกำลังไฟฟ้า

ภาพทางขวายังแสดงผลเฉลย การจ่ายกำลัง ไฟฟ้าอย่างประหยัดดังนี้ P_1 = 450.000000, P_2 = 324.999997 และ P_3 = 199.999998 ซึ่งตรงตามความต้องการกำลังไฟฟ้า 975 MW ที่กำหนด



รูปที่ 7 การใช้ MATLAB กับปัญหาการจ่ายกำลังไฟฟ้าเดียวกัน

และด้วยปัญหาเดียวกันนี้ หากนำไปหาผลเฉลยด้วยซอฟต์แวร์สำเร็จ รูป MATLAB ดังรูปที่ 7 ก็ได้ผลที่ตรงกัน วัตถุเชิงพันธุกรรมที่สร้างขึ้น ยังสามารถประ ยุกต์ใช้กับการหาผลเฉลยของสมการไม่เชิงเส้นที่เกิดขึ้นในปัญหาการไหลของภาระ [10] ดังต่อไปนี้



รูปที่ 8 แผนภาพเส้นเดี่ยวของระบบกำลังไฟฟ้า 2 บัส

ผลเฉลยของปัญหาการไหลของภาระตามรูป ที่ 8 คือแรงดันและกำลังรีแอกทีฟที่บัสหมายเลข (2) ระบบ สมการไม่เชิงเส้นของกำลังเชิงซ้อนแต่ละบัสคือ

$$\begin{bmatrix} S_1 \\ S_2 \end{bmatrix} = \left(V_1 \begin{bmatrix} Y_{11} & Y_{12} \\ 0 & 0 \end{bmatrix}^* + V_2 \begin{bmatrix} 0 & 0 \\ Y_{21} & Y_{22} \end{bmatrix}^* \right) \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}^*$$

เมื่อ $Y_{11} = -j2, Y_{12} = Y_{21}$ และ $Y_{22} = Y_{11}$ กำลังเชิงซ้อนเหล่านี้จะถูกใช้เป็นฟังก์ชันเป้าหมาย สำหรับการคำนวณหาค่า V_2 ที่จะทำให้ $\mathrm{Re}(S_2) = P_2$ เมื่อ $P_2 = \mathrm{Re}(S_{G2}) - \mathrm{Re}(S_{D2}) = -0.75$

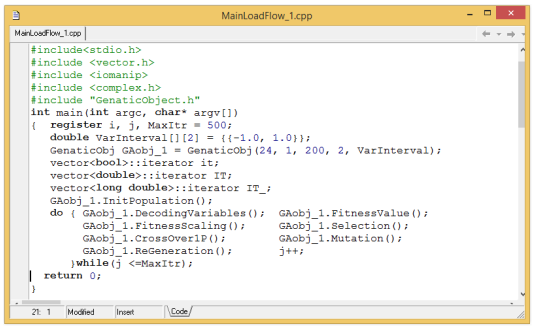
รูปที่ 9 แสดงการใช้ GeneticObj วัตถุเชิงพันธุกรรมของบทความ เพื่อหาผลเฉลยดังกล่าว(ตัดคำสั่งพิมพ์ผล) ผลการใช้วัตถุเชิงพันธุกรรม GeneticObj ในรูปที่ 10 คือตัวอย่างสมาชิกของประชากรรุ่นแรกที่ได้จากการสุ่มทั้งสิ้น 200 โครโมโซมตามโปรแกรมในรูปที่ 9 แต่สำหรับในรูปที่ 11 แสดงประชากรในรุ่นสุดท้ายรอบที่ 500 ที่แสดงออกถึงการลู่เข้าของประชากร เพื่อใช้ในการกำหนดผลเฉลย ของปัญหาแรงดันและกำลังเชิงซ้อน ที่คำนวณได้ด้วยการใช้วัตถุเชิงพันธุกรรมนี้คือ

$$V_2 = 0.927017 - j0.375018 = 1\angle -22.0254$$

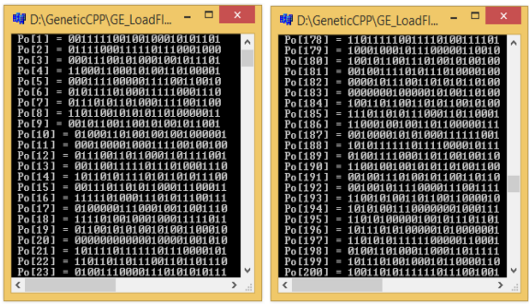
$$S_2 = -0.750001 + j0.145951$$

$$S_1 = 0.750001 + j0.145951$$

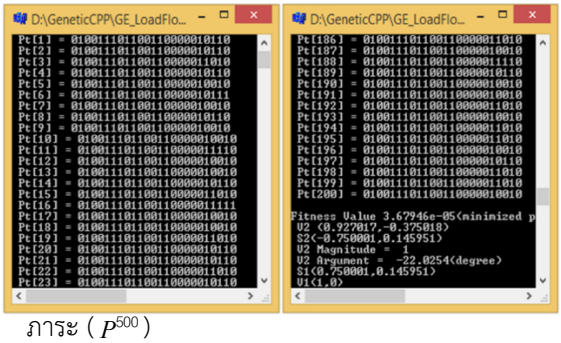
ตามรายละเอียดของผลการพิมพ์ในรูปที่ 12



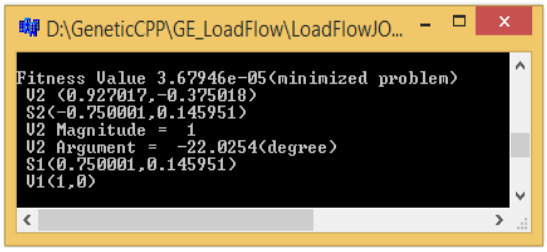
รูปที่ 9 การใช้วัตถุเชิงพันธุกรรมกับปัญหาการไหลของภาระ



รูปที่ 10 ผลการใช้ GeneticObj ในการไหลของภาระ(P^0)

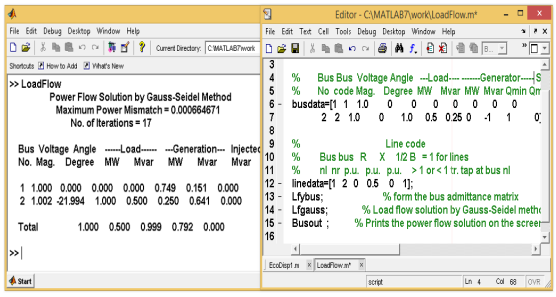


ภาระ (P^{500})



รูปที่ 12 ผลเฉลยการไหลของภาระโดย GeneticObj

เพื่อเปรียบเทียบผลการทำงานจะใช้ซอฟต์แวร์ สำเร็จรูป MATLAB คำนวณผลเฉลยของปัญหาการไหลของภาระเดียวกันนี้



รูปที่ 13 ผลเฉลยการไหลของภาระโดย MAT LAB

ซึ่งเห็นได้ชัดว่า ผลเฉลยของแรงดันและกำลังเชิงซ้อนโดยใช้ระเบียบวิธีเกาส์–ไซเดล (Gauss–Seidel method) ของโปรแกรม MATLAB ให้ผลการคำนวณผลเฉลยมีค่าใกล้เคียงกันกับการใช้วัตถุเชิงพันธุกรรม GeneticObj ของบทความนี้

สรุปผลการทดลอง

วัตถุเชิงพันธุกรรม GeneticObj ที่สร้างขึ้นถึงแม้จะเริ่มต้นการคำนวณผลเฉลยเชิงตัวเลข ด้วยการสุ่มผลเฉลยในรูปประชากรเริ่มต้น แทนการกำหนด ค่าผลเฉลยเริ่มต้นในวิธีทำซ้ำ (iteration) ของระเบียบวิธีเชิงตัวเลข แต่ก็ยังต้องอาศัยการกำหนดย่านของผลเฉลยที่กำหนดโดยค่าขอบเขตล่าง l_i และขอบเขตบน u_i สำหรับถ้อยแถลงทางคณิตศาสตร์ที่ตั้งไว้ ในวิธีทำซ้ำของระเบียบวิธีเชิงตัวเลข การกำหนดค่าผลเฉลยเริ่มต้น จะส่งผลต่อการลู่เข้าของวิธีทำซ้ำสู่ผลเฉลย สำหรับการกำหนดย่านของผลเฉลยให้วัตถุเชิงพันธุกรรม ที่ไม่มีผลเฉลยของปัญหาตกอยู่ภายในย่านนั้น จะทำให้ขั้นตอนวิธีเชิงพันธุกรรมวิวัฒนาการประชากร เข้าสู่ค่าใดค่าหนึ่งในย่านที่ไม่ใช่ผลเฉลย นำไปสู่ความล้มเหลวของวัตถุเชิงพันธุกรรมที่สร้างขึ้น แต่ในอีกทางหนึ่งวัตถุเชิงพันธุกรรมไม่ต้องการวิธีวิเคราะห์ทางคณิตศาสตร์ขั้นสูงเพื่อการกำหนดวิธีทำซ้ำเช่นในระเบียบวิธีเชิงตัวเลข ใช้เพียงการสุ่มผลเฉลยในย่าน และการวิวัฒนาการตามขั้นตอนเชิงพันธุกรรม โดยการดำเนินงานที่ผ่านมา ใช้เพียงการสุ่มเทียมของภาษาการโปรแกรมซีพลัสพลัส ทำให้ง่ายในการพัฒนาวัตถุเชิงพันธุกรรมและประสบผลสำเร็จในการหาผลเฉลย เนื่องจากปัญหาการจ่ายกำลังไฟฟ้าอย่างประหยัด มีเงื่อนไขบังคับที่ทำให้สามารถกำหนดย่านผลเฉลยได้ง่าย ขณะที่ปัญหาการไหลของภาระมีปกติในการใช้ระบบค่าต่อหน่วย (per-unit) เป็นข้อมูลการคำนวณอยู่แล้ว ทำให้ผลเฉลยของปัญหาถึงแม้จะเป็นจำนวนเชิงซ้อน แต่ขนาดของผลเฉลยซึ่งคือแรงดันในแต่ละบัสของระบบกำลังไฟฟ้า จะมีค่าใกล้เคียง 1 หน่วย ขณะที่มุมเฟสเรเดียนต์ (radiant) จะอยู่ในย่านค่าบวกและค่าลบที่แคบมาก ทำให้การกำหนดย่านของผลเฉลยทำได้ง่ายเช่นกัน

เอกสารอ้างอิง

[1] Belegundu, A. D. and Chandrupatla, T. R. Optimization Concepts and Applications in Engineering. Cambridge University Press ;2011; 318 - 324.

[2] Duman, S., Özturk, A., Dosoglu, M. K., and Tosun, S. Economic Dispatch by Using Different Crossover Operators of Genetic Algorithm. IU - JEEE, 2010; vol.10, Issue 19; 1173-1183.

[3] Ghiasi, M., Rashtchi, V. and Hoseini, S. H. Optimum Location and Sizing of Passive Filters in Distribution Networks Using Genetic Algorithm. In: Emerging Technologies, 2008. ICET 2008. 4th International Conference on. IEEE 2008; 162-166.

[4] Gjylapi, D. and Kasëmi, V. The Genetic Algorithm for Finding The Maxima of Single-variable Functions. International Journal Of Engineering And Science 2014; , Vol.4, Issue 3: PP 46-54.

[5] Kaur, A., Singh, H. P. and Bhardwaj, A. Analysis of Economic Load Dispatch Using Genetic Algorithm. IJAIEM 2014 , vol.3, Issue 3: 240 – 246.

[6] Kim, H., Samann, N., Shin, D., and Ko, B. Jang, G. and Cha, J. A New Concept of Power Flow Analysis. JEET 2007 , vol. 2 no. 3312 – 319.

[7] Lubkeman, D.L., Ghosh, A. and Zhu, J. Object-oriented Programming for Power Engineering Education. *Proc.* SSST '93., Twenty-Fifth South eastern Symposium on,Tuscaloosa: 1993;, 69-73.

[8] Mastorakis, N.E. Solving Non-linear Equations via Genetic Algorithms. *Proc.* the 6th WSEAS Int, Lisbon: 2005; 24 -28.

[9] Nasiruzzaman, A.B.M. and Rabbani, M.G. Implementation of Genetic Algorithm and Fuzzy Logic in Economic Dispatch Problem. *Proc.* ICECE 2008, Dhaka:2008: 360 -365.

[10] Toso, R.F. and Resende, M. G. A C++ Appli cation Programming Interface for Biased Random-key Genetic Algorithms. Optimization Methods and Software 2015; Vol.30, Issue 181-93.