รายการอ้างอิง

Arunwatanamongkol P., "A Computer Program For Heat Exchanger Network Design Using Match Patterns Approach", Master's Thesis, Chulalongkorn Universuty, (1992).

Calandranis, J. and Stephanopoulos, G., "Structural Operability Analysis of Heat Exchanger Network", Chem. Eng Res. Des., 64, 347 (1986).

Cerda, J., A.W. Westerberg, D. Mason and B. Linnhoff, "Minimum Utility Usage in Heat Exchanger Network Synthesis - A Transportation Problem", Chem.Eng. Sci., 38 (3), 373 (1983a).

_____. and A.W. Westerberg, "Synthesizing Heat Exchanger Networks having Restricted Stream/Stream Matches using Transportation Problem Formulations", Chem. Eng. Sci., 38(10), 1723 (1983b).

_____., M.R. Galli, N. Camussi and M.A. Isla, "Synthesis of Flexible Heat Exchanger Networks-I. Convex Networks", Comp. Chem. Engng., 14(2), 197-211 (1990a).

_____. and M.R. Galli, "Synthesis of Flexible Heat Exchanger Networks-II. Nonoconvex Networks with Latge Temperature Variations", Comp. Chem. Engng., 14(2), 213-225 (1990b).

Chen, B. , J. Shen, Q. Sun, snd S. Hu, "Development of Expert System for Synthesis of Heat Exchanger Networks", Comput. Chem. Engng., 13(11/12), 1221 (1989).

Floudas, C.A., A.R. Ciric and I.E. Grossmann, "Automatic Synthesis of Optimum Heat Exchanger Network Configurations", AIChE J., 32(2), 276 (1986).

_____. and I.E. Grossmann, "Synthesis of Flexible Heat Exchanger Networks for Multiperiod Operation", Comput. Chem. Engng., 10(2), 153 (1986).

_____. and I.E. Grossmann, "Synthesis of Flexible Heat Exchanger Networks with Uncertain Flowrates and Temperatures", Comput. Chem. Engng., 11(4), 319-336 (1987).

Hohmann, E. C., Optimum Networks for Heat Exchanger. Ph.D. Dissertation, University of South California, (1971).

Jezowski, J. and E. Hahne, "Heat Exchanger Network Synthesis by a Depth-First Search Method-A Case Study", Chem. Eng. Sci., 41(12), 2989 (1986).

Linnhoff, B., et al., User Guide on Process Integration for the Efficient Use of Energy, Inst. of Chem. Engrs. UK (1982).

_____. and Ahmad, S., "Cost Optimum Heat Exchanger Network s-1. Minimum Energy and Capital Using Simple Models for Capital Cost", Comput. Chem.Engng, 14(7), 729-750 (1990).

_____. and J.R.Flower, "Synthesis of Heat Exchanger Networks-I. Systematic Generation of Energy Optimal Networks", AlChE J., 24(4), 633-642 (1978a).

_____. and J.R.Flower, "Synthesis of Heat Exchanger Networks-II. Evolutiionary Generation of Networks with Various Criteria of Optimality", AlChE J., 24(4), 642-654 (1978b).

_____. and Hindmarsh, E., "The Pinch Design Method for Heat Exchanger Networks", Chem.Eng. Sci., 38(5), 745-763 (1983).

_____. and E. Kotjabasakis, "Design of Operable Heat Exchanger Networks", First U.K. National Heat Transfer Conference, I. Chem. E. Symp. Ser. No.86, Vol 1, 599 (1984).

_____. and E. Kotjabasakis, "Downstream Paths for Operable Process Design", CEP, 82(5), 23 (1986).

Marselle, D.F., M.Morai and D.F. Rudd, "Design of Resilient Processing Plants-II: Design and Control of Energy and Management System", Chem. Eng. Sci., 37(2), 259 (1982).

Masso, A.H. and D.F. Rudd, "The Synthesis of System Designs-II. Heuristic Structuring", AIChE J., 15, 10-18 (1969).

Mehta. C.D. and L.T. Fan, "Heat Exchanger Network Synthesis: A Knowledge Engineering Approach", Paper presented at AIChE annual meeting, New York City, November (1987).

Papoulias, S.A. and I.E. Grossmannn "A Structual Optimizatioon Approach in Process Synthesis-II. Utility Systems", Comput. Chem. Engng., 7, 707-722 (1983).

Pehler, F.A. and Y.A. Liu, "Efficiency and Costing: Second Law Analysis of Processes", ACS Symposium Series No.135, ACS, Washington D.C. (1983).

Pho, T.K. and L.Lapidus, "Topics in Computer Aides Design-II. Synthesis of Optimal Heat Exchanger Networks by Tree Search Algorithms", AIChE J., 19(6), 1182 (1973).

Ponton, J.W. and R.A.B. Donaldson, "A Fast Method for Synthesis of Optimal heat Exchanger Networks", Chem. Eng. Sci., 29, 2375-2377 (1974).

Rathore, R.N.S. and G.J. Powers, "A Forward Branching Scheme for the Synthesis of Energy Recovery System", Ind. Eng. Chem. Proc. Des. Dev., 14, 175 (1975).

Saboo, A.K. and M. Morari, "Design of Resilient Processing Plants-IV: Some New Results on Heat Exchanger Network Synthesis", Chem. Eng. Sci., 39(3), 579 (1984).

_____. and M. Morari and D.C. Woodcock, "Design of Resilient Plants-VIII: A Resilience Index for Heat Exchanger Networks", Chem. Eng. Sci., 40, 1553 (1985).

_____., M.Morari and R.D. Colberg, "Resilience Analysis of Heat Exchanger Networks-I. Temperature Dependent Heat Capacities", Comput. Chem. Engng., 11 (4), 399 (1987a).

_____., M.Morari and R.D. Colberg, "Resilience Analysis of Heat Exchanger Networks-II. Stream Splits and Flowrate Variations", Comput. Chem. Engng., 11(5), 457 (1987b).

Su, J.L.and R.L.Motard, "Evolutionary Synthesis of Heat Exchanger Networks", Comput. Chem. Engng., 8, 67 (1984).

Swaney, R.E.and I.E.Grossmann, "An Index for Operational Flexibility in Chemical Process Design, Part I&II", AIChE J., 31(4), 621 (1985).

Wistler, A.M. "Heat Exchanger as Money Makers", Pet. Refiner., 27, 83 (1948).

Smith, R., "Heat Exchanger Network and Utilities: Capital and Total Cost Targets", Chemical Process Design, McGraw-Hill, New York, (1995).

Wongsri, M., Resilient Heat Exchanger Network Design, Docteral Dissertation, Washington University, (1990).

ภาคผนวก ก

ตัวอย่างคลาส กระสวนการจับคู่ และโค้ดโปรแกรม

ก.1 ตัวอย่างการออกแบบคลาสกระแสและคลาสโหนด

สำหรับคลาสกระแสที่ออกแบบไว้มีลักษณะดังนี้

```
class Streams
{ public:
   short Stat;
   float TsupplyMin,TsupplyNom,TsupplyMax,Ttarget,TinterIn,TinterOut;
   float FCp,FCpX,FCpN,Q,Qx;
   float Dstb1,Dstb2,DstbW,h;
   Streams() {}
  ~Streams() {}
};
```

จากนิยามของคลาสกระแสจะเห็นว่า สำหรับกระแสใด ๆ ก็ตามจะต้องมีข้อมูลที่จำเป็น คือ อุณหภูมิขาเข้าและขาออก ช่วงของอุณหภูมิขาเข้าที่แปรปรวน ค่าผลคูณระหว่างอัตราการไหลและความจุความร้อนจำเพาะ เป็นต้น ส่วนความแปรปรวนและความร้อนของกระแสนั้นสามารถคำนวณได้ในภายหลังจากค่าดังกล่าว

เมื่อจะทำการสร้างวัตถุกระแสร้อนและกระแสเย็นโดยให้มีชื่อว่า HotStreams และ ColdStreams ตามลำดับ สามารถสร้างได้โดยใช้คำสั่ง

Streams HotStreams, ColdStreams;

กระแสร้อนและกระแสเย็นที่สร้างขึ้น จะมีตัวแปรเหมือนกับที่นิยามไว้ในคลาสกระแสทุกประการ กระแสร้อน (HotStream) และกระแสเย็น (ColdStream) นี้ ข้อมูลจะเป็นอิสระไม่เกี่ยวข้องกันแต่อย่างใด การอ้างถึงข้อมูลเช่น อุณหภูมิขาเข้าปกติของกระแสร้อนนั้น ก็สามารถเขียนได้เป็น HotStreams.TsupplyNom เป็นต้น การอ้างถึงข้อมูลตัวอื่นๆ ก็ทำในลักษณะทำนองเดียวกัน ต่อไปนี้ คือ นิยามของคลาสโหนด หรือคลาสของเครื่องแลกเปลี่ยนความร้อน

```
class NodeType
{ public:
  Streams *HotStreams,*ColdStreams;
  short  Side,StatH,StatC,Life;
  unsigned short Parent,Split,K,I,J,Child;
  unsigned short KC,IC,JC;
  float ThIn,ThOut,TcIn,TcOut,Q,Qacc;
  MatchObject() {}
 ~MatchObject() {}
  void SetStreams(int NoSh, int NoSc);
};
```

เนื่องจากเราใช้คลาสโหนดเป็นตัวเก็บข้อมูลกระแสที่แลกเปลี่ยนความร้อนในสภาวะต่างๆ ข่ายงานแลกเปลี่ยนความร้อนจะประกอบด้วยโหนดหลายๆ โหนด หรืออาจมองได้ว่าโหนดๆ หนึ่งก็คือ เครื่องแลกเปลี่ยนความร้อน 1 เครื่อง ดังนั้น นอกจากที่คลาสโหนดจะมี

คลาสกระแสเป็นสมาชิกอยู่ด้วยแล้ว ยังมีพารามิเตอร์ที่ใช้เก็บข้อมูลดังที่จะกล่าวต่อไปนี้ คือ

Life - ใช้บอกสถานะของโหนดว่าสามารถสร้างโหนดอื่นๆ ต่อจากโหนดนี้ได้ไหม หรือกล่าวอีกนัยหนึ่ง โหนดนี้ยังมีชีวิตอยู่หรือตายไปแล้ว โดยกำหนดค่าสถานะเป็น 1 และ -1

Parent - แสดงถึงเลขโหนดที่เป็นโหนดแม่

Child - บอกถึงจำนวนโหนดของโหนดลูก

K,I,J - เก็บค่าแพทเทิร์นกระสวนการจับคู่ระหว่างกระแสร้อนและเย็นของโหนดปัจจุบัน

KC, IC, JC - เก็บค่าแพทเทิร์นกระสวนการจับคู่ระหว่างกระแสร้อนและเย็นของโหนดลูกล่าสุด

Side - ใช้บอกลักษณะการแลกเปลี่ยนความร้อนของกระสวนการจับคู่ที่ใช้ว่า อยู่ปลายอุณหภูมิสูง (มีค่าเป็น 1) หรือปลายอุณหภูมิต่ำ (มีค่าเป็น -1) เพื่อให้ความสะดวกต่อการแสดงผลทางกราฟฟิก

SetStreams - เป็นฟังก์ชันซึ่งใช้สร้างคลาสกระแสของโหนดขึ้นมาใหม่ต่อจาก (หรือเป็นลูกของ) โหนดปัจจุบัน โดยจะต้องส่งค่าของตัวแปรต่างๆ ของคลาสกระแสจากโหนดปัจจุบันทั้งหมดไปให้โหนดที่สร้างใหม่

ThIn, ThOut - อุณหภูมิขาเข้าและขาออกของกระแสร้อน

TcIn, TcOut - อุณหภูมิขาเข้าและขาออกของกระแสเย็น

Q - ความร้อนที่ใช้ของเครื่องแลกเปลี่ยนความร้อน

Qacc - ความร้อนหลงเหลือสะสมหลังจากการแลกเปลี่ยนความร้อนระหว่างกระแสที่ใช้ในการสร้างหน่วยยูทิลิตี

สำหรับการสร้างวัตถุของโหนดขึ้นมาตามนิยาม (ในที่นี้ให้มีชื่อว่า Node) ก็สามารถทำได้โดยใช้คำสั่งดังนี้

```
NodeType  Node;
```

การสร้างวัตถุกระแสร้อน, วัตถุกระแสเย็น และวัตถุโหนดนี้ จะเป็นตัวแปรชนิดอาเรย์โดยจองหน่วยความจำแบบไดนามิก ซึ่งจะใช้หน่วยความจำเท่าที่จำเป็นสำหรับปัญหาหนึ่งๆ เท่านั้น รูปแบบการสร้างวัตถุตามชนิดของคลาสโดยเป็นไดนามิกอาเรย์นี้ มีรูปแบบทั่วไป ดังนี้

```
Class *Object;

Object = new Class[No];
```

เมื่อ No คือ จำนวนของวัตถุที่เราต้องการสร้าง สำหรับไดนามิกอาเรย์ของวัตถุนี้ เมื่อได้ทำการสร้างแล้ว จะต้องลบออกทุกครั้งเมื่อสิ้นสุดการทำงานของโปรแกรม โดยใช้คำสั่ง

```
Delete[]  Object;
```

ก.2 ตัวอย่างโค้ดโปรแกรมกระสวนจับคู่

กระสวนการจับคู่ทั้งหมดนี้ จะถูกรวมเข้าด้วยกันในฟังก์ชันหนึ่งๆ และจะเรียงลำดับตามการออกแบบเหนือจุดพินช์ หรือใต้จุดพินช์ ซึ่งได้กล่าวมาแล้วในบทที่ 3 โดยถ้าเป็นการออกแบบข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์จะใช้ฟังก์ชัน MatchPatternA แต่ถ้าเป็นแบบใต้จุดพินช์จะใช้ฟังก์ชัน MatchPatternB สำหรับการออกแบบข่ายงานแบบยืดหยุ่นเหนือจุดพินช์ใช้ฟังก์ชัน MatchPatternAR แต่ถ้าเป็นใต้จุดพินช์จะใช้ฟังก์ชัน MatchPatternBR โดยโค้ดของ

ฟังก์ชัน MatchPatternA และ MatchPatternB จะเหมือนกัน ฟังก์ชัน MatchPatternAR และ MatchPatternBR จะเหมือนกัน ต่างกันเฉพาะที่ลำดับของแพทเทิร์นตามการจับคู่เหนือ หรือใต้จุดพินช์เท่านั้น

การอ้างอิงถึงโหนดใดๆ นั้น ในโปรแกรมจะมีตัวแปรอยู่สามตัวด้วยกัน คือ สำหรับโหนดที่โปรแกรมกำลังอ้างถึงอยู่ในปัจจุบัน จะใช้ตัวแปร n, สำหรับหมายเลขโหนดล่าสุดที่โปรแกรมสร้าง จะใช้ตัวแปร ns และสำหรับการอ้างถึงโหนดแม่ของโหนดปัจจุบันนั้นๆ จะใช้ตัวแปร No

เมื่อโปรแกรมเรียกใช้แพทเทิร์นใดแพทเทิร์นหนึ่ง สมมติกำลังใช้แพทเทิร์นที่ 2 (A[H]) ออกแบบข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์ (ในที่นี้ฟังก์ชัน MatchPatternA จะทำงาน) โปรแกรมจะส่งค่าแพทเทิร์น (*k*), หมายเลขของกระแสร้อน (*i*), หมายเลขของกระแสเย็น (*j*) และหมายเลขโหนดในการอ้างอิงมาที่ฟังก์ชัน ฟังก์ชันจะไปทำงานกรณีของแพทเทิร์น A[H] หรือ Case 2 ตามค่า *k* ที่ส่งมา จากนั้นจะทดสอบอุณหภูมิ และอัตราการไหลตามตามเงื่อนไขของแพทเทิร์น A[H] ถ้าได้ตามเงื่อนไขก็จะทำการคำนวณ และตั้งค่าความร้อนและอุณหภูมิของกระแสใหม่ สร้างโหนดใหม่ด้วยฟังก์ชัน ConstructNode(n,No) แล้วจะส่งค่า 1 กลับไปยังส่วนของโปรแกรมที่เรียกใช้ (Return 1) แต่ถ้าไม่สามารถใช้เงื่อนไขของแพทเทิร์นนี้ได้ ก็จะส่งค่ากลับไปเป็น 0 (Return 0)

สำหรับการออกแบบข่ายงานแบบไม่ยืดหยุ่นจะมีตัวแปรเพิ่มขึ้นมา เช่นค่า *S* (พารามิเตอร์ความยืดหยุ่นของกระแส), *E* (พารามิเตอร์ความยืดหยุ่นของกระแส) และ *Dstb* (ความแปรปรวน) เป็นต้น โดยใช้เงื่อนไขในการพิจารณาตามที่ได้กล่าวมาแล้วในบทที่ 3

ในที่นี้จะยกตัวอย่างโค้ดของกระสวนการจับคู่แพทเทิร์น A[H] ทั้งแบบยืดหยุ่นและไม่ยืดหยุ่น สำหรับแพทเทิร์นอื่นๆ ก็จะมีลักษณะการพิจารณาคล้ายๆ กัน ฟังก์ชันกระสวนการจับคู่นี้ จะถูกเรียกใช้โดยฟังก์ชัน Mach() ซึ่งจะได้กล่าวต่อไปภายหลัง

ก.2.1 กระสวนการจับคู่แบบไม่ยืดหยุ่นแพทเทิร์น A[H]

```
// For Non-Resilient Match Patterns
int MatchStreams::MatchPatternA(int &i,int &j,int &k,int &ns,int &No)
{ float T;
 switch(k)
  {
       case 1:  // AH
                :

       case 2:  // A[H]
        if(Node[n].HotStreams[i].TinterOut-Node[n].ColdStreams[j].TinterIn>=Tmin)
          if(Node[n].HotStreams[i].Q <= Node[n].ColdStreams[j].Q && Node[n].
            ColdStreams[j].FCp >= Node[n].HotStreams[i].FCp)
              { T = Node[n].ColdStreams[j].TinterIn + Node[n].HotStreams[i].Q /
                    Node[n].ColdStreams[j].FCp;
                if(Node[n].HotStreams[i].TinterIn-T >= Tmin)
                 { n=ns; n++; ConstructNode(n,No);
```

```
                    Node[n].HotStreams[i].Stat = 1;
                    Node[n].ColdStreams[j].Q  -= Node[n].HotStreams[i].Q;
                    Node[n].Q                  = Node[n].HotStreams[i].Q;
                    Node[n].HotStreams[i].Q   = 0;
                    Node[n].Side               = 1;
                    Node[n].StatH              = i;
                    Node[n].StatC              = j;
                    Node[n].ThOut              = Node[n].HotStreams[i].TinterOut;
                    Node[n].TcOut              = T;
                    Node[n].ColdStreams[j].TinterIn = T;
                    return(1);
                  }
                }
          return(0);
        case 3:
          .
          .
          .
        case 9:
    }
}
```

ก.2.2 กระสวนการจับคู่แบบยืดหยุ่นแพทเทิร์น A[H]

```
//For Resilient Match Patterns
int MatchStreams::MatchPatternAR(int &i,int &j,int &k,int &ns,int &No)
{ float ΔT,T,S_CH,S_HC,E_CH,E_HC;
 S_CH = (Node[n].HotStreams[i].FCp-Node[n].ColdStreams[j].FCp)*(Node[n].HotStreams[i].
        TinterIn-Node[n].HotStreams[i].TinterOut);
 S_HC = (Node[n].ColdStreams[j].FCp-Node[n].HotStreams[i].FCp)*(Node[n].ColdStreams[j].
        TinterOut-Node[n].ColdStreams[j].TinterIn);
 if(Node[n].HotStreams[i].TinterOut-Node[n].ColdStreams[j].TinterIn <=
  Node[n].HotStreams[i].TinterIn-Node[n].ColdStreams[j].TinterOut)
        ΔT = Node[n].HotStreams[i].TinterOut-Node[n].ColdStreams[j].TinterIn;
 else
        ΔT = Node[n].HotStreams[i].TinterIn-Node[n].ColdStreams[j].TinterOut;
 E_CH = Node[n].ColdStreams[j].FCp*(ΔT-Tmin);
 E_HC = Node[n].HotStreams[i].FCp*(ΔT-Tmin);
 if(FCpStatus==0) S_CH = S_HC = 0;

 switch(k)
 {      case 1:  // A[H]
         if(Node[n].HotStreams[i].Dstb_W <= E_CH + S_CH)
          if(Node[n].HotStreams[i].Qx <= Node[n].ColdStreams[j].Q &&
            Node[n].ColdStreams[j].FCpMin >= Node[n].HotStreams[i].FCpMin)
                { T = Node[n].ColdStreams[j].TinterIn + Node[n].HotStreams[i].Qx /
                      Node[n].ColdStreams[j].FCpMin;
                 if(Node[n].HotStreams[i].TinterIn-T >= Tmin )
                  { n = ns;  n++;  ConstructNode(n,No);
                    Node[n].HotStreams[i].Stat  = 1;  Node[n].HotStreams[i].Q    = 0;
```

```
                    Node[n].ColdStreams[j].Q  -= Node[n].HotStreams[i].Qx;
                    Node[n].Q                 = Node[n].HotStreams[i].Qx;
                   if(Node[n].HotStreams[i].TinterOut==Node[n].ColdStreams[j].TinterIn
                     && Node[n].HotStreams[i].Dstb2/Node[n].HotStreams[i].FCpMin
                      ==Node[n].ColdStreams[j].Dstb1/Node[n].ColdStreams[j].FCpMin)
                  Node[n].ColdStreams[j].Dstb1  += Node[n].HotStreams[i].
                  Dstb1-Node[n].HotStreams[i].Dstb2;
                else
                  Node[n].ColdStreams[j].Dstb1  += Node[n].HotStreams[i].Dstb1
                    + Node[n].HotStreams[i].Dstb2;
                Node[n].HotStreams[i].Dstb1  = 0;
                Node[n].ColdStreams[j].Qx    = Node[n].ColdStreams[j].Q +
                  Node[n].ColdStreams[j].Dstb1;
                Node[n].Side                 = 1;
                Node[n].StatH                = i;
                Node[n].StatC                = j;
                Node[n].ThOut                = Node[n].HotStreams[i].TinterOut;
                Node[n].TcOut = Node[n].ColdStreams[j].TinterIn = T;
                return(1);
              }
            }
            return(0);
            case 2:  // B[C]
            case 4:  // B[H]
        }
    }
```

ก.3 ตัวอย่างโค้ดของการออกแบบโครงสร้างข่ายงาน

การออกแบบโครงสร้างข่ายงาน (ส่วนของ Matching จากรูปที่ 4.1) นั้นจะมีฟังก์ชันหลัก คือ MatchCenter ซึ่งโปรแกรมภายนอกจะติดต่อกับฟังก์ชันนี้เท่านั้น โดยโปรแกรมภายนอกที่เรียกใช้จะส่งข้อมูลของกระแสร้อนและกระแสเย็น จำนวนกระแส ΔT_{min} อุณหภูมิพินช์ ฯลฯ เข้ามา MatchCenter จะสร้างชุดของกระแสร้อน และกระแสเย็นอันใหม่ไว้ใช้เองเพื่อป้องกันการแก้ไข และเปลี่ยนแปลงข้อมูลของกระแสร้อนและเย็นของชุดเดิมที่มีอยู่ ชุดกระแสร้อนและกระแสเย็นขึ้นมาใหม่นี้จะอยู่ภายในคลาส MatchStreams (ทุกฟังก์ชันของการออกแบบหาโครงสร้างข่ายงานจะอยู่ภายในคลาส MatchStreams ทั้งหมด) สร้างโดยใช้ฟังก์ชัน ConstructStreams(HotStreams,ColdStreams) จากนั้นฟังก์ชัน MatchCenter จะใช้ข้อมูลที่ได้ตรวจสอบว่าข่ายงานมีจุดพินช์หรือไม่ และจะเลือกใช้ฟังก์ชันได้ตามความเหมาะสม

สมมติว่าในการออกแบบข่ายงานเครื่องแลกเปลี่ยนความร้อนข่ายงานหนึ่ง จากการคำนวณพบว่ามีอุณหภูมิพินช์ สำหรับการออกแบบข่ายงานเหนือจุดพินช์ ในที่นี้จะตรวจสอบค่า $Q_{H,min}$ มีค่ามากกว่า 0 หรือไม่ ถ้าใช่ ก็จะทำการตั้งค่าอุณหภูมิขาเข้าและออกของกระแสใหม่ตามอุณหภูมิพินช์ และคำนวณความแปรปรวน (ถ้าเป็นข่ายงานแบบยืดหยุ่น) โดยใช้ฟังก์ชัน SetStreamA() จากนั้นก็จะสร้างโหนด 0 ขึ้นโดยฟังก์ชัน ConstructNode(0) ใช้ฟังก์ชัน Macth() เพื่อหาโครงสร้างของข่ายงานที่เป็นไปได้ ส่วนที่เหลือจะเป็นการเรียกใช้ฟังก์ชันการแสดงผล

แต่ถ้าเป็นการออกแบบข่ายงานใต้จุดพินช์ โปรแกรมจะตรวจสอบค่า $Q_{C,min}$ และปรับค่าอุณหภูมิขาเข้าและขาออกของกระแสใหม่ ด้วยฟังก์ชัน SetStreamB() หลังจากนั้นก็จะสร้าง

โหนด 0 ด้วยฟังก์ชัน ConstructNode(0) เช่นกัน เสร็จแล้วก็จะเรียกฟังก์ชัน Match() เพื่อหาโครงสร้างของข่ายงานต่อไป

สำหรับข่ายงานที่ออกแบบไม่มีอุณหภูมิพินช์ การปรับตั้งค่าและการคำนวณจะใช้ฟังก์ชันการออกแบบเหนือจุดพินช์ หรือใต้จุดพินช์อย่างใดอย่างหนึ่งก็ได้ เนื่องจากหลักการพิจารณาจะใช้อย่างเดียวกัน แต่จะเรียกใช้ฟังก์ชันครั้งเดียว เนื่องจากไม่ต้องแบ่งข่ายงานออกเป็นข่ายงานย่อยสองข่ายงานตามอุณหภูมิพินช์อีก ในที่นี้ใช้การออกแบบที่เหนือจุดพินช์แทน

จะสังเกตได้ว่า ไม่ว่าจะเป็นการออกแบบเหนือจุดพินช์ หรือใต้จุดพินช์ก็ตาม จะสร้างโหนด 0 ไว้เป็นโหนดแรกเสมอ ลำดับชุดของโหนดต่อมาจะเป็นของข่ายงานเหนือจุดพินช์หรือใต้จุดพินช์ ก็ขึ้นกับว่าขณะนั้นโปรแกรมคำนวณที่เหนือจุดพินช์หรือใต้จุดพินช์นั่นเอง โดยอาเรย์ของโหนดจะใช้พื้นที่หน่วยความจำเดียวกันในการเก็บข้อมูล เพื่อเป็นการประหยัดหน่วยความจำ

การทำงานของฟังก์ชัน Match() จะทำงานโดยเรียกลำดับแพทเทิร์นที่ k ออกมาก่อน แล้วจึงเรียกกระแสร้อน และกระแสเย็นมาตามลำดับ เมื่อได้คู่ลำดับครบจะเรียกใช้ฟังก์ชันกระสวนการจับคู่ตามความเหมาะสม จากนั้นการวนรอบจะให้ครบรอบกระแสเย็นก่อน เมื่อครบแล้วจึงวนรอบกระแสร้อน แล้ววนรอบแพทเทิร์นในที่สุด ถ้าวนรอบแพทเทิร์นครบแล้วและสถานะของโหนดกลับมาอยู่ที่โหนด 0 จะถือเป็นการสิ้นสุดการทำงานของฟังก์ชัน Match()

ฟังก์ชันอื่นๆ ที่ได้แสดงไว้ในที่นี้ด้วยก็คือ

- ฟังก์ชัน Heater() และ Cooler() ฟังก์ชันสองฟังก์ชันนี้ มีไว้เพื่อสร้างฮีตเตอร์และคูลเลอร์ในข่ายงาน จะใช้เมื่อกระแสร้อนและเย็นได้ทำการแลกเปลี่ยนความร้อนจนความร้อนของชุดกระแสร้อนหรือเย็นชุดใดชุดหนึ่งหมดไป โดยตรวจสอบด้วยค่า CheckNoStream ในที่นี้กระแสที่ความร้อนยังไม่หมดจะให้สถานะเป็น 0 และกระแสที่ความร้อนยังไม่หมดจะให้สถานะเป็น 1 CheckNoStream จะรวมค่าดังกล่าว ไว้เปรียบเทียบกับจำนวนกระแสทั้งหมด (ในชุดใดชุดหนึ่ง) ถ้าค่าทั้งสองเท่ากันจึงหยุดการวนรอบหาโครงสร้างข่ายงานและเรียกใช้ฟังก์ชัน Heater() และ Cooler() ดังกล่าว
- ฟังก์ชัน SetIJK() มีไว้สำหรับปรับตั้งค่าเลขลำดับของกระแสร้อน (i), กระแสเย็น (j) และกระสวนการจับคู่ (k) ในกรณีที่ได้สร้างโหนดใหม่ได้แล้วและกำลังจะทำการสร้างโหนดอื่นต่อไป หรือกรณีที่โหนดที่กำลังใช้งานได้ตายไป (ไม่สามารถทำการค้นหาคำตอบต่อไปได้อีก) และต้องทำการค้นหาโดยวิธีย้อนโหนดจากล่างขึ้นบน
- ฟังก์ชัน CheckSolution() ฟังก์ชันนี้ใช้ตรวจสอบโครงสร้างข่ายงานที่หาได้ เกิดซ้ำกับโครงสร้างเดิมที่เคยมีมาก่อนหรือไม่ โดยตรวจสอบจากอุณหภูมิขาเข้า และขาออกของกระแสร้อนและกระแสเย็นของเครื่องแลกเปลี่ยนความร้อนทุกๆ เครื่อง (โหนด) ในข่ายงาน

ต่อไปนี้ เป็นโค้ดของโปรแกรมในส่วนการออกแบบโครงสร้างข่ายงานที่ได้กล่าวมาทั้งหมด

```
void MatchStreams::MatchCenter(Streams *HotStreams, Streams *ColdStreams, int nh, int nc,
float tmin, float tpinch, float qh, float qc, float u, int R, float TpMn, float TpMx, int S, int p)
{
 Tmin        = tmin;
 TpinchH     = tpinch;
 TpinchC     = tpinch-tmin;
 QhMin       = qh;
 QcMin       = -qc;
 ResStatus   = R;
 TpinchMin   = TpMn;
 TpinchMax   = TpMx;
 FCpStatus   = S;
 NoSh        = nh;
 NoSc        = nc;
 U           = u;
 split       = 0;
 ConstructStreams(HotStreams,ColdStreams);

 int i,j,k,c,Check=0,dig=7,NSplit=0,nsp=0;
 char ch[20];
 if((QhMin>0 && p==1)|| p==3)
  { Check=1;mouse_hide_cursor();
    for(i=0;i<=(No_Node-1)/2;i++)
         Solution[i]=0;
    SetStreamsA(); ConstructNode(0);
    split=0;Match(1);
    i=1;k=0;
```

```
while(Solution[i]!=0)
 { k++; i++; }
NoSolutionA=NoSplitA=k;
split=1;Split(1);split=0;
i=1;k=0;
while(Solution[i]!=0)
      {k++; i++;}
NoSolutionA=k;


for(;;)
 { for(;;)
    { c = HotKey(' ');
         aa:
         switch(c)
              { case 73:
                     if(i==k) continue;
                     i++;if(i>k) i=k;goto ab;
                case 81:
                     if(i==1)continue;
                     i--;if(i<1) i=1;goto ab;
                case 27:
                     if(QcMin>0) goto s;
                      else goto ss;
                case 23:
                     Area=0;Information(Solution[i],1);
              }
        m_check=ON;
```

```
            if(mouse_called())
                { m_check=OFF;
                 if(m_col>=4&&m_col<=24&&m_row>=58&&m_row<=76)
                  { pictureX(1);pictureX();c=27;goto aa;}
                   if(m_col>=594&&m_col<=614&&m_row>=58&&m_row<=76)
                       {pictureLT(594,58,-1);pictureLT(594,58);c=81;goto aa;}
                   if(m_col>=615&&m_col<=635&&m_row>=58&&m_row<=76)
                       {pictureRT(615,58,-1);pictureRT(615,58);c=73;goto aa;}
                   if(m_col>=24&&m_col<=44&&m_row>=58&&m_row<=76)
                       {pictureI(24,58,-1);pictureI(24,58);c=23;goto aa;}
                  }
              }
           }//end for 2
         ab:  mouse_hide_cursor(); ShowMatch(1,Solution[i]);
          PrintTemp(1,Solution[i]); Area=0;ShowMatchA(Solution[i]);
          mouse_show_cursor();
        }//end for 1
    }//end main if
s:
if(p==3) goto ss;
if(QcMin>0 && p==2)
  { mouse_hide_cursor();
    SetStreamsB(); ConstructNode(0);
    for(i=0;i<=(No_Node-1)/2;i++) Solution[i]=0;
    split=0;Match(2);
    i=1;k=0;
    while(Solution[i]!=0)
```

```
    {k++; i++;}
NoSolutionB=NoSplitB=k;
split=1;Split(2);split=0;
i=1;k=0;
while(Solution[i]!=0)
    {k++; i++;}
NoSolutionB=k;

for(;;)
 { for(;;)
   { c = HotKey(' ');
    bb:
    switch(c)
          { case 73:
             if(i==k) continue;
             i++;if(i>k) i=k; goto ba;
           case 81:
             if(i==1) continue;
             i--;if(i<1) i=1; goto ba;
           case 27:
             goto ss;
           case 23:
             Area=0;Information(Solution[i],2);
          }
   m_check=ON;
```

```
            if(mouse_called())
                { m_check=OFF;
                 if(m_col>=4&&m_col<=24&&m_row>=58&&m_row<=76)
                  { pictureX(1);pictureX();c=27;goto bb;}
                   if(m_col>=594&&m_col<=614&&m_row>=58&&m_row<=76)
                        {pictureLT(594,58,-1);pictureLT(594,58);c=81;goto bb;}
                   if(m_col>=615&&m_col<=635&&m_row>=58&&m_row<=76)
                        {pictureRT(615,58,-1);pictureRT(615,58);c=73;goto bb;}
                   if(m_col>=24&&m_col<=44&&m_row>=58&&m_row<=76)
                        {pictureI(24,58,-1);pictureI(24,58);c=23;goto bb;}
                }
        }//end for 2
           ba:  mouse_hide_cursor();  ShowMatch(2,Solution[i]);
                PrintTemp(2,Solution[i]);  Area=0;  ShowMatchB(Solution[i]);
     }//end for 1
   }//end main if
 ss:
}

int MatchStreams::Match(int &Select,int NoS)
{ int i,SameSolution,j,k,s,CheckNoStream;
 int No,ns,np,nk;
 static int NoSolution;
 int K0,I0,J0;
 char ch[10];
 if(split==0)  { NoSolution=1;n=ns=No=0; }
 else  { ns=n;No=n; }
```

```
K0=I0=J0=0=s=CheckNoStream=0;
if(NoShP<=0 && NoScP>=1)
  { ns=1; Heater(ns); goto end; }
if(NoScP<=0 && NoShP>=1)
  { ns=1; Cooler(ns); goto end; }
if(ResStatus==1)  nk=9;
  else   nk=4;
x:
if(n==0 && split==1) goto end;
for(k=K0;k<=nk;k++)
 { for(i=I0;i<NoSh;i++)
    { if(Node[n].HotStreams[i].Stat!=0 || Node[n].HotStreams[i].Q==0) continue;
      for(j=J0;j<NoSc;j++)
       { if(Node[n].ColdStreams[j].Stat!=0 || Node[n].ColdStreams[j].Q==0) continue;
         if(ResStatus==1)
          { if(Select==1)
                  s = MatchPatternA(i,j,k,ns,No);
            else
                  s = MatchPatternB(i,j,k,ns,No);
           if(ns>=No_Node-10)
                  { outtextxy(20,459,"Memory not enough ,set new No. of searching
                        node (Alt-p)"); getch(); goto end;
                  }
                }
         else if(ResStatus==2)
                { if(Select==1)
                   s = MatchPatternAR(i,j,k,ns,No);
```

```
            else
              s = MatchPatternBR(i,j,k,ns,No);
            if(ns>=No_Node-10)
              { outtextxy(20,459,"Memory not enough ,set new No. of searching
                  node (Alt-p)"); getch(); goto end;
              }
          }
if(s==1)
  {     ns++;
        if(k==1 && ResStatus==1) { ns++; No=n-1; }
        if(k>1) Node[n].Parent  = No;
        Node[n].Pattern = k;
        if((k>1&&ResStatus==1) || ResStatus==2)
          { Node[n].K    = k;
            Node[n].I    = i;
            Node[n].J    = j;
          }
        Node[No].Child++;
        Node[No].KC   = k;
        Node[No].IC   = i;
        Node[No].JC   = j;
        No = n;
        K0 = 1; I0 = J0 = 0;
        CheckNoStream=0;
        for(int g=0;g<NoSh;g++)
          if(Node[n].HotStreams[g].Stat>=1) CheckNoStream++;
        if(CheckNoStream==NoShP) Heater(ns);
```

```
                    CheckNoStream=0;
                    for(g=0;g<NoSc;g++)
                      if(Node[n].ColdStreams[g].Stat>=1) CheckNoStream++;
                    if(CheckNoStream==NoScP) Cooler(ns);
                    goto x;
                }
            } //end for j
        } //enf for i
        I0 = J0 = 0;
    } //enf for k
Node[n].Life = -1;
if(n!=0)
  { if(Node[n].Life == -1 && Node[n].Child == 0)
      { ns--; Node[Node[n].Parent].Child--;
        if(Node[n].Pattern==1 && ResStatus==1)
          { Node[Node[n].Parent].Life = -1;
            No = Node[Node[n].Parent].Parent; n = No;
            ns--; Node[n].Child--;
          }
        else  No = Node[n].Parent;   n = No;
      }
    else
      { if(Node[n].Pattern==1 && ResStatus==1)
          No = Node[Node[n].Parent].Parent;
        else
          No = Node[n].Parent;  n = No;
      }
```

```
    SetIJK(I0,J0,K0,CheckNoStream); goto x;
  }
goto end;

y: SameSolution=0;
if(NoSolution>=2)
  SameSolution = CheckSolutions(ns,NoSolution-1);
if(SameSolution==0)
 { Solution[NoSolution] = ns;
   Node[ns].Split = 0;
   if(split==1)
     Node[ns].Split = StreamsSplit+1;
   if(NoS==NoSolution)
     { NoSolution++;goto end;}
   NoSolution++;
 }
else
 { np = ns;
   if(Node[np].Pattern==1 && ResStatus==1||(Node[ns].StatC<0 &&
     Node[ns].StatH<0))
     { if(Node[Node[np].Parent].K==0)
        { np = Node[np].Parent;ns--; }
         np = Node[Node[np].Parent].Parent;
         ns-=2;
        }
   else
        { np = Node[np].Parent;ns--;}
```

```
      Node[np].Child--;
      n = No = np;
      SetIJK(I0,J0,K0,CheckNoStream);
      goto x;
     }
   if(((Node[n].Pattern==1 || Node[n].Pattern==9) && ResStatus==1)||
    (Node[n].Pattern==4 && ResStatus==2))
     { No = Node[Node[n].Parent].Parent; n = No;}
   else
     { No = Node[n].Parent; n = No;}
   end:
   if(Solution[1]==0) n=0;
   n=ns;
   if(NoSolution>=2)
     { if(split==0 && NoS==NoSolution-1)  return 1;
       else if(split==0 && NoS!=NoSolution-1)  return 0;
     }
   if(Solution[1]>0)  return 1;
   else  return 0;
}

void MatchStreams::Heater(int &ns)
{  for(int j=0;j<NoSc;j++)
    { if(Node[ns].ColdStreams[j].Q > 0 && Node[ns].ColdStreams[j].Stat==0)
       { ns++;  ConstructNode(ns,ns-1);  Node[ns-1].Child++;
         Node[ns].Parent    = ns-1;
         Node[ns].ThOut     = 0;
```

```
        Node[ns].TcOut    = Node[ns].ColdStreams[j].TinterOut;
        Node[ns].StatH    = -1;
        Node[ns].StatC    = -1-j;
        Node[ns].Q        = Node[ns].ColdStreams[j].Q;
        Node[ns].ColdStreams[j].Stat =  2;
        Node[ns].Side     =  1;
        Node[ns].K        =  0;
        Node[ns].I        =  0;
        Node[ns].J        =  j;
          }
    }
}

void MatchStreams::Cooler(int &ns)
{ for(i=0;i<NoSh;i++)
   { if(Node[ns].HotStreams[i].Q > 0 && Node[ns].HotStreams[i].Stat==0)
      { ns++;  ConstructNode(ns,ns-1);  Node[ns-1].Child++;
        Node[ns].Parent   = ns-1;
        Node[ns].TcOut    = 0;
        Node[ns].ThOut    = Node[ns].HotStreams[i].TinterOut;
        Node[ns].StatH    = -1-i;
        Node[ns].StatC    = -1;
        Node[ns].Q        = Node[ns].HotStreams[i].Q;
        Node[ns].HotStreams[i].Stat =  2;
        Node[ns].Side     =  1;
        Node[ns].K        =  0;
        Node[ns].I        =  i;
        Node[ns].J        =  0;
```

```
      }
    }
}

void MatchStreams::SetIJK(int &I0, int &J0, int &K0, int &CheckNoStream)
{if(Node[n].JC<=NoScP-2 && NoScP>=2)
   { J0 = Node[n].JC+1; I0 = Node[n].IC;
     K0 = Node[n].KC; CheckNoStream = 0;
   }
 else if(Node[n].JC>=NoScP-1 && Node[n].IC <= NoShP-2)
   { J0 = 0; I0 = Node[n].IC+1;
     K0 = Node[n].KC; CheckNoStream = 0;
   }
 else if(Node[n].JC>=NoScP-1 && Node[n].IC>=NoShP-1 && Node[n].KC<nk)
   { J0 = 0; I0 = 0;
     K0 = Node[n].KC+1; CheckNoStream = 0;
   }
}

int MatchStreams::CheckSolutions(int ns,int s,int ab)
{int no,np,nl1,nl2,c,z,si=1;
 c=nl1=nl2=z=0;
 no = ns;

 while( no != 0)
   { nl1++; no = Node[no].Parent; }
 for(int i=si;i<=s;i++)
```

```
{ np = Solution[i];
 while( np != 0 )
  { nl2++; np = Node[np].Parent;}
   if(nl1==nl2)
    { no=ns;
     while( no != 0 )
            { np = Solution[i];
             while( np != 0 )
              { if(Node[no].I==Node[np].I && Node[no].J==Node[np].J)
               if(Node[no].ThOut==Node[np].ThOut &&
                  Node[no].TcOut==Node[np].TcOut)
                   { c++; z = 1;
                    break;
                   }
                  np = Node[np].Parent;
              }
             if(z==0) break;
             no = Node[no].Parent;
             z=0;
            }
      }
       if(c==nl1)   return(1);
      }
      nl2 = 0; c = 0;
 }//end for i
 return(0);
}
```

ก.4 ตัวอย่างโค้ดโปรแกรมส่วนการหาค่า ΔT_{min} ที่ออปติมัม

สำหรับการหาค่า ΔT_{min} ที่ออปติมัมนี้ ต้องใช้ข้อมูลอุณหภูมิ, อัตราการไหลของกระแส รวมทั้งสมการค่าใช้จ่ายทางพลังงานความร้อนด้วย การทำงานของโปรแกรมส่วนนี้ได้แบ่งออกเป็นสามส่วนหลักๆ คือ ฟังก์ชัน OptimizeTmin(), Inputh() และ Area() สำหรับฟังก์ชัน OptimizeTmin() จะเป็นฟังก์ชันหลัก ถ้าผู้ใช้ให้ค่า U ไม่คงที่ ฟังก์ชันนี้จะเรียกใช้ฟังก์ชัน Inputh () ซึ่งจะให้ตารางรับค่าสัมประสิทธ์การถ่ายเทความร้อนผ่านฟิล์ม การคำนวณค่าใช้จ่ายจะแบ่งออกเป็นสองประเภท คือ ค่าใช้จ่ายด้านพลังงานของข่ายงาน และค่าใช้จ่ายในการสร้างข่ายงาน โดยปริมาณความร้อนที่ใช้ ณ ΔT_{min} ค่าต่างๆ จะคำนวณโดยวิธีตารางปัญหา ซึ่งฟังก์ชัน OptimizeTmin() จะเรียกใช้ฟังก์ชันย่อยของวิธีตารางปัญหาซึ่งมีอยู่แล้ว (ดูหัวข้อ ก.5 วิธีตารางปัญหา)

ส่วนการคำนวณค่าใช้จ่ายในการสร้างข่ายงาน ฟังก์ชัน OptimizeTmin() จะเรียกใช้ฟังก์ชัน Area() โดยฟังก์ชันนี้จะส่งค่ากลับเป็นพื้นที่ของเครื่องแลกเปลี่ยนความร้อนมาให้ เมื่อได้ข้อมูลของปริมาณความร้อนและพื้นที่แลกเปลี่ยนความร้อนของข่ายงาน ที่ ΔT_{min} ค่าหนึ่งๆ แล้ว ก็จะทำการคำนวณค่าใช้จ่ายออกมา จากนั้นฟังก์ชัน OptimizeTmin() จะเพิ่มค่า ΔT_{min} ไปเรื่อยๆ จนถึงช่วงที่กำหนด ก็จะทำการตรวจสอบว่าค่าใช้จ่ายรวม ณ ΔT_{min} เท่ากับเท่าไหร่ มีค่าต่ำที่สุด จากนั้นจะแสดงค่า ΔT_{min} ที่ออปติมัมให้ผู้ใช้ทราบ สำหรับฟังก์ชันทั้งหมดที่แสดงนี้จะอยู่ในคลาสการออกแบบข่ายงานเบื้องต้น (Class PreAnalysis) ต่อไปนี้เป็นตัวอย่างของฟังก์ชันในการหาค่า ΔT_{min} ที่ออปติมัมทั้งสามฟังก์ชัน คือ Inputh(), Area() และ OptimizeTmin()

```
int PreAnalysis::Inputh(Streams* HotStreams, Streams* ColdStreams, int &hs, float *TEMPH)
{ int i,nNoSh+NoSc;,x,py,y,No=NoSh+NoSc+3,*temph = new int[No];
 char ch[15];
 for(i=0;i<No;i++) temph[i]=0;
while(k==0)
{ //Input data******
 if(gscanf_last_key==72||gscanf_last_key==80||gscanf_last_key==-
    1||gscanf_last_key==9||gscanf_last_key==13||mouse_called())
   {  if(gscanf_last_key==-1||gscanf_last_key==72){ci--;if(ci<1) ci=1;}
     else if(gscanf_last_key==9||gscanf_last_key==13||gscanf_last_key==80)
         { ci++;if(ci>=No) ci--; }
     else
          { for(i=1;i<=2*NoSh;i++) HotStreams[i-1].h=TEMPH[i];
            for(i=2*NoSh+1;i<=No-3;i++)  ColdStreams[i-(NoSh+1)].h=TEMPH[i];
           HUs.h = TEMPH[No-2];    CUs.h = TEMPH[No-1];
          }
        }
     }
 }//end while
 delete[] temph;  return k;
}

float PreAnalysis::Area(Streams *H,Streams *C,int us)
{ int nh,nc,m,n,HglCStatus,k=0;
 float *TH,*TC,thi,tho,tci,tco,hh,hc,Q,Qh,Qc,QphH,QphC,area,tln;
 if(QhMin>0) nh=2*(NoSh+1)+1;   else nh=2*NoSh+1;
 if(QcMin<0) nc=2*(NoSc+1)+1;   else nc=2*NoSc+1;
```

```
hh=hc=Qh=Qc=area=0;  TH  = new float[nh];  TC  = new float[nc];
for(int i=1;i<=NoTh;i++) TH[i]=Tmh[i]; for(int j=1;j<=NoTc;j++) TC[j]=Tmc[j];
i=NoTh;j=NoTc;
if(QhMin>0)
  {HUs.Tout = HUs.Tin - QhMin/HUs.FCp ; TH[i+1]= HUs.Tin; TH[i+2]= HUs.Tout;  }
if(QcMin<0)
 {CUs.Tout = CUs.Tin - QcMin/CUs.FCp ;  TC[j+1]= CUs.Tin; TC[j+2]= CUs.Tout;  }
ArangeData(TH,nh-1); ArangeData(TC,nc-1);
for(i=1;i<nh-1;i++)
        if(TH[i]==TH[i+1])
                {for(j=i;j<nh-k;j++) TH[j]=TH[j+1];   i++;k++;}
nh=nh-k;  k=0;
for(i=1;i<nc-1;i++)
        if(TC[i]==TC[i+1])
                { for(j=i;j<nc-k;j++)   TC[j]=TC[j+1];   i++;k++;}
nc=nc-k;  m=n=2;tci=TC[n-1];tco=TC[n];thi=TH[m-1];tho=TH[m];
for(j=0;j<NoSh;j++)
        if(thi>=H[j].Ttarget && tho<=H[j].TsupplyNom) Qh+=H[j].FCp*(tho-thi);
        if(thi>=HUs.Tout && tho<=HUs.Tin&&QhMin>0) Qh+=HUs.FCp*(tho-thi);
hh+=Qh;
for(j=0;j<NoSc;j++)
        if(tci>=C[j].TsupplyNom && tco<=C[j].Ttarget) Qc+=C[j].FCp*(tco-tci);
        if(tci>=CUs.Tin && tco<=CUs.Tout&&QcMin<0) Qc+=CUs.FCp*(tco-tci);
hc+=Qc;
do
{if(hh>hc) { Q=Qc; HglCStatus=1; tho=thi+((tho-thi)/Qh)*(hc-hh+Qh);}
 else if(hh<hc) { Q=Qh; HglCStatus=-1; tco=tci+((tco-tci)/Qc)*(hh-hc+Qc);}
```

```
else if(hh==hc) { Q=Qh; HglCStatus=0;}
if((tho-tco)/(thi-tci)<=0) return 0;
if((tho-tco)!=(thi-tci)) tln=(tho-tco-thi+tci)/log((tho-tco)/(thi-tci));
   else tln = tho-tco;
if(us==1) area+=Q/(U*tln);
else { QphH=0;
                for(j=0;j<NoSh;j++)
                if(thi>=H[j].Ttarget && tho<=H[j].TsupplyNom)
                        QphH+=H[j].FCp*(tho-thi)/H[j].h;
                if(thi>=HUs.Tout && tho<=HUs.Tin&&QhMin>0)
                        QphH+=HUs.FCp*(tho-thi)/HUs.h;
                QphC=0;
                for(j=0;j<NoSc;j++)
                if(tci>=C[j].TsupplyNom && tco<=C[j].Ttarget)
                        QphC+=C[j].FCp*(tco-tci)/C[j].h;
                if(tci>=CUs.Tin && tco<=CUs.Tout&&QcMin<0)
                        QphC+=CUs.FCp*(tco-tci)/CUs.h;
                Q=QphH+QphC;  area+=Q/tln;
        }
if(HglCStatus==1)
        { thi=tho;tho=TH[m]; n++;tci=tco;tco=TC[n];Qh=Qc=0;
                for(j=0;j<NoSh;j++)
                 if(thi>=H[j].Ttarget && tho<=H[j].TsupplyNom)
                   Qh+=H[j].FCp*(tho-thi);
                if(thi>=HUs.Tout && tho<=HUs.Tin&&QhMin>0)
                 Qh+=HUs.FCp*(tho-thi);
                for(j=0;j<NoSc;j++)
```

```
                if(tci>=C[j].TsupplyNom && tco<=C[j].Ttarget)
                  Qc+=C[j].FCp*(tco-tci);
                if(tci>=CUs.Tin && tco<=CUs.Tout&&QcMin<0)
                 Qc+=CUs.FCp*(tco-tci);
                hc+=Qc;
        }
else if(HglCStatus==-1)
        { tci=tco;tco=TC[n]; m++;thi=tho;tho=TH[m];Qh=Qc=0;
                for(j=0;j<NoSh;j++)
                 if(thi>=H[j].Ttarget && tho<=H[j].TsupplyNom) Qh+=H[j].FCp*(tho-thi);
                 if(thi>=HUs.Tout && tho<=HUs.Tin&&QhMin>0) Qh+=HUs.FCp*(tho-thi);
                 hh+=Qh;
                 for(j=0;j<NoSc;j++)
                  if(tci>=C[j].TsupplyNom && tco<=C[j].Ttarget) Qc+=C[j].FCp*(tco-tci);
                 if(tci>=CUs.Tin && tco<=CUs.Tout&&QcMin<0)Qc+=CUs.FCp*(tco-tci);
        }
else if(HglCStatus==0)
        { tci=tco;thi=tho;m++;n++;tco=TC[n];tho=TH[m];Qh=Qc=0;
                 for(j=0;j<NoSh;j++)
                  if(thi>=H[j].Ttarget && tho<=H[j].TsupplyNom)  Qh+=H[j].FCp*(tho-thi);
                 if(thi>=HUs.Tout && tho<=HUs.Tin&&QhMin>0) Qh+=HUs.FCp*(tho-thi);
                 hh+=Qh;
                 for(j=0;j<NoSc;j++)
                  if(tci>=C[j].TsupplyNom && tco<=C[j].Ttarget)  Qc+=C[j].FCp*(tco-tci);
                 if(tci>=CUs.Tin && tco<=CUs.Tout&&QcMin<0) Qc+=CUs.FCp*(tco-tci);
                 hc+=Qc;
        }
```

```
}while(m<nh&&n<nc);
 delete[] TH;  delete[] TC;  return area;
}

int PreAnalysis::OptimizeTmin(Streams *Sh,Streams *Sc,int &round,int &hs,float *TEMPH)
{float Th=HUs.Tin,Wh=HUs.FCp,Tc=CUs.Tin,Wc=CUs.FCp;
 float hucost=HUCost,cucost=CUCost,acapcost=ACapCost,bcapcost=BCapCost,it=interest;
 static int k=1;  char cc[20];
 int i,kk=0,n=10,u=0,ustatus=1,c,ch,ci=1,yr=years,*tempo= new int[11];
 for(i=0;i<11;i++) tempo[i]=0;
 while(kk==0)
        { c=HotKey(' ');
              a: m_check=ON;  // Input data ****
              if(c==13 && ci==11)  {   k++;u=1;  } //Select U
                    else if(c==13 && ci==12) //enter + Ok
                     {      HUs.Tin  = Th;      HUs.FCp  = Wh;   interest = it;
                            CUs.Tin  = Tc;      CUs.FCp  = Wc;   years    = yr;
                            HUCost   = hucost;  CUCost   = cucost;
                            ACapCost = acapcost; BCapCost = bcapcost;
                            if(k%2!=1)
                                  { if(Inputh(Sh,Sc,hs,TEMPH)==1)
                                     { round+=2; hs=1;}   ustatus=-1;
                                  }
                            else   {U = InputU();round++;ustatus=1;}
                            kk=1; continue;
                            }
                     }
```

```
                }
            }
          }//end while
 delete[] tempo;
 int t,NoShP,NoScP,Nu=0,rnd=0,cj=1;
 float area,An,TminOld=Tmin,TminOpt,ymin=0,ymax=0,*Cost;
 if(interest<=0||years<=0) An=1;
   else An = interest*pow(1+interest,years)/(pow(1+interest,years)-1);
for(;;)
 { Cost = new float[xe+1];
        for(t=xo;t<=xe;t++)
         { Tmin=t;Nu=0;ArangeData(T,Def,l,1);
           Temp(T,Df,l);Interval(); CalQ();   ArangeData(T,Def,l,3);
        // above  pinch
                NoShP=NoSh+1;
                for(i=0;i<NoSh;i++)  if(Sh[i].TsupplyNom<=Tpinch) NoShP--;
                if(HUs.Tin<=Tpinch&&QhMin>0) NoShP--;
                Nu+=NoShP;NoScP=NoSc+1;
                for(i=0;i<NoSc;i++) if(Sc[i].Ttarget<=Tpinch-Tmin) NoScP--;
                if(CUs.Tout<=Tpinch-Tmin&&QcMin<0) NoScP--;
                Nu+=NoScP;
        // below pinch
                NoShP=NoSh+1;
                for(i=0;i<NoSh;i++) if(Sh[i].Ttarget>=Tpinch) NoShP--;
                if(HUs.Tout>=Tpinch&&QhMin>0) NoShP--;
                Nu+=NoShP;NoScP=NoSc+1;
                for(i=0;i<NoSc;i++) if(Sc[i].TsupplyNom>=Tpinch-Tmin) NoScP--;
```

```
                if(CUs.Tin>=Tpinch-Tmin&&QcMin<0) NoScP--;
                Nu+=NoScP;    Nu-=2;    area = Area(Sh,Sc,ustatus);
                Cost[t] = HUCost*QhMin-CUCost*QcMin+An*Nu*(ACapCost+
                        BCapCost*area/Nu);
        }
 Tmin = TminOld;  ArangeData(T,Def,l,1); Temp(T,Df,l);
 Interval(); CalQ(); ArangeData(T,Def,l,3); ymax=1.1*ymax;ymin=0.8*ymin;
 if(TminOpt>=2)
        {       moveto(2*w/xe,h-((Cost[2]-ymin)/(ymax-ymin))*100*h/ye);
                for(i=xo;i<=xe;i++)  lineto(i*w/xe,h-((Cost[i]-ymin)/(ymax-ymin))*100*h/ye);
        }
        for(;;) {ch = HotKey(' '); if((ch==9||ch==75||ch==77)&&cj==2) cj--;  }
 r:delete[] Cost;
}
end:delete[] Cost;
if(Tmin!=TminOpt)
 if(AcceptTminOpt(Tmin,TminOpt)==1)
        { Tmin=TminOpt;  Temp(T,Df,l); Interval();   CalQ();ArangeData(T,Def,l,3); }
return 1;
}
```

ก.5 ตัวอย่างโค้ดของวิธีตารางปัญหา

ตารางปัญหามีความสำคัญที่จะทำให้รู้ว่า ค่าพลังงานความร้อนที่จะต้องให้เพิ่มเข้าไปในข่ายงาน ($Q_{H,min}$) หรือความร้อนที่จะต้องดึงออกมาจากข่ายงาน ($Q_{C,min}$) หรืออุณหภูมิพินช์มีค่าเท่าใด ปริมาณความร้อนที่คำนวณได้จะนำไปคำนวณหาค่าใช้จ่ายทางพลังงานได้

ฟังก์ชันของวิธีตารางปัญหาจะอยู่ในคลาสการออกแบบเบื้องต้น โดยโปรแกรมหลักจะต้องส่งวัตถุกระแสเย็นและกระแสร้อนมาให้ โดยมีฟังก์ชัน SetData() รับข้อมูล จากนั้นโปรแกรมหลักจะเรียกใช้ฟังก์ชัน Solve() ซึ่งเป็นฟังก์ชันหลักของวิธีตารางปัญหา ฟังก์ชัน Solve() จะเรียกใช้ฟังก์ชันย่อยอีกสี่ฟังก์ชันตามลำดับอีก ดังต่อไปนี้

1. ArangeData(T,Def,l,1);

2. Temp(T,Df,l);

3. Interval();

4. CalQ();

5. ArangeData(T,Def,l,3);

การเรียกใช้ฟังก์ชัน ArangeData() ครั้งแรกนี้ เป็นการจัดเรียงข้อมูล เช่นอุณหภูมิกระแสจากน้อยไปมาก ลำดับที่สองจะเรียกฟังก์ชัน Temp() จะเป็นการเตรียมการสร้างช่วงอุณหภูมิ (Temperature interval) ฟังก์ชันที่สาม Interval() จะสร้างช่วงอุณหภูมิตามข้อมูลที่ได้เตรียมไว้ และฟังก์ชัน CalQ() เป็นการหาค่าพลังงานที่ใช้น้อยที่สุดของหน่วยยูทิลิตี และหาอุณหภูมิพินช์ออกมา และลำดับสุดท้ายสำหรับการเรียกฟังก์ชัน ArangeData() ครั้งที่สองนี้ เป็นการตั้งลำดับ

ข้อมูลของกระแสให้กลับเป็นเหมือนเดิมก่อนการคำนวณ ต่อไปนี้เป็นตัวอย่างโค้ดของโปรแกรมส่วนของวิธีตารางปัญหาดังกล่าว

```
void PreAnalysis::SetData(Streams *Hot,Streams *Cold,int s)
{ int i; l=0;
 Tmh[0]=Tmc[0]=NULL;
 T[0]=Def[0]=Df[0]=NULL;

 NoTc=0;
 for(i=1;i<=NoSc;i++)
 {      l++; NoTc++;
            if(s==0) //Normal condition
                  { T[l] = TmpC[i].input = Tmc[NoTc] = Cold[i-1].TsupplyNom ;
                    FCpSc[i] = Cold[i-1].FCp;
                  }
            else if(s==1) //Max. heat load
                  { T[l] = TmpC[i].input = Tmc[NoTc] = Cold[i-1].TsupplyMin ;
                    if(ResStatus==2&&FCpStatus==1) FCpSc[i] = Cold[i-1].FCpX;
                          else FCpSc[i] = Cold[i-1].FCp;
                  }
            else if(s==2)//Min. heat load
                  { T[l] = TmpC[i].input = Tmc[NoTc] = Cold[i-1].TsupplyMax ;
                    if(ResStatus==2&&FCpStatus==1)  FCpSc[i] = Cold[i-1].FCpN;
                          else FCpSc[i] = Cold[i-1].FCp;
                  }
           Sc[NoTc]  = i;Df[l]=Def[l]= 0 ;  l++; NoTc++;
```

```
                T[l] = TmpC[i].output = Tmc[NoTc] = Cold[i-1].Ttarget ;Df[l]=Def[l]= 0 ;
        }

NoTh=0;
for(i=1;i<=NoSh;i++)
{       l++;  NoTh++;
                if(s==0)//Normal condition
                        { T[l] = TmpH[i].input = Tmh[NoTh] = Hot[i-1].TsupplyNom;
                          FCpSh[i] = Hot[i-1].FCp;
                        }
                else if(s==1) //Max. heat load
                        { T[l] = TmpH[i].input = Tmh[NoTh] = Hot[i-1].TsupplyMax;
                          if(ResStatus==2&&FCpStatus==1) FCpSh[i] = Hot[i-1].FCpX;
                                else FCpSh[i] = Hot[i-1].FCp;
                        }
                else if(s==2) //Min. heat load
                        { T[l] = TmpH[i].input = Tmh[NoTh] = Hot[i-1].TsupplyMin;
                          if(ResStatus==2&&FCpStatus==1) FCpSh[i] = Hot[i-1].FCpN;
                                else FCpSh[i] = Hot[i-1].FCp;
                        }
                Def[l]=Df[l]= 1 ; l++;  NoTh++;
                T[l] = TmpH[i].output = Tmh[NoTh] = Hot[i-1].Ttarget;
                Sh[NoTh]  = i;Def[l]=Df[l]=1;
 }
 T[l+1]=Def[l+1]=Df[l+1]=NULL;
}
```

```
void PreAnalysis::Solve(float t,int W)
{ Tmin = t;
 ArangeData(Tmh,Sh,NoTh,2);
 ArangeData(Tmc,Sc,NoTc,2);
 ArangeData(T,Def,l,1);
 Temp(T,Df,l);Interval(); CalQ();
 ArangeData(T,Def,l,3);

 if(HUs.FCp!=0)HUs.Tout = HUs.Tin - QhMin/HUs.FCp ;
 if(CUs.FCp!=0)CUs.Tout = CUs.Tin - QcMin/CUs.FCp ;
 FCpStatus=W;
}

void PreAnalysis::ArangeData(float *W,int *C,int N,int Select)
{int i,z,k;
 float M,P;
 // max to min
 if(Select==1)
        { z=1;
          while(z)
                {z=0;
                        for(i=1;i<=N-1;i++)
                        { if(C[i]==1&&C[i+1]==1&&W[i]>W[i+1]) continue;
                                if(C[i]==0&&C[i+1]==0&&W[i]>W[i+1]) continue;
                                if(C[i]==1&&C[i+1]==0&&W[i]-Tmin>=W[i+1]) continue;
                                if(C[i]==0&&C[i+1]==1&&W[i]+Tmin>=W[i+1]) continue;
                                if(W[i]==W[i+1]&&C[i]==0&&C[i+1]==1) continue;
```

```
                    if(W[i]>=W[i+1]) continue;
                    M = W[i];    k = C[i];    P = Def[i];
                    W[i]=W[i+1]; C[i]=C[i+1]; Def[i]=Def[i+1];
                    W[i+1]=M;    C[i+1]=k;    Def[i+1]=P;
                    z=1;
                }
            }
        }
// min to max
 if(Select==2)
        { z=1;
            while(z)
            { z=0;
                for(i=1;i<=N-1;i++)
                    { if(W[i]<=W[i+1]) continue;
                     M = W[i];    k = C[i];    //P = Def[i];
                     W[i]=W[i+1]; C[i]=C[i+1]; //Def[i]=Def[i+1];
                     W[i+1]=M;    C[i+1]=k;    //Def[i+1]=P;
                     z=1;
                    }
            }
        }
//max to min
 for(i=1;i<=N;i++) Df[i]=C[i];
 if(Select==3)
        { z=1;
          while(z)
```

```
                {z=0;
                        for(i=1;i<=N-1;i++)
                        {  if(W[i]>=W[i+1]) continue;
                                M = W[i];     k = C[i];
                                W[i]=W[i+1];  C[i]=C[i+1];
                                W[i+1]=M;     C[i+1]=k;
                                Df[i]=C[i];Df[i+1]=C[i+1];
                                z=1;
                        }
                }
        }
}

void PreAnalysis::ArangeData(float *W,int N)
{int i,z;
 float M;
// min to max
                z=1;
                while(z)
                { z=0;
                        for(i=1;i<=N-1;i++)
                                { if(W[i]<=W[i+1]) continue;
                                 M = W[i];
                                 W[i]=W[i+1];
                                 W[i+1]=M;
                                 z=1;
                                }
```

```
                }
}

void PreAnalysis::Temp(float *w,int *c,int N)
{Ts[0]=NULL;s=0;
 int i,j;
 for(i=1;i<=N;i++)
        { if(c[i]==1)
                { for(j=1;j<=N;j++)
                    if(i!=j&&c[j]==0&&w[i]-Tmin==w[j]||(c[j]==1&&w[i]==w[j])) c[j]=-1;
                    s++;Ts[s]=w[i];s++;Ts[s]=w[i]-Tmin;c[i]=-1;
                }
         else if(c[i]==0)
                { for(j=1;j<=N;j++)
                    if(i!=j&&c[j]==1&&w[i]+Tmin==w[j]||(c[j]==0&&w[i]==w[j])) c[j]=-1;
                    s++;Ts[s]=w[i]+Tmin;s++;Ts[s]=w[i];c[i]=-1;
                }
        }
}

void PreAnalysis::Interval()
{int i,j,k;
 float WCpSct,WCpSht;
 WF *WFH = new WF[NoSh+1];
 WF *WFC = new WF[NoSc+1];
 for(i=0;i<=NoSh;i++) WFH[i].WFS(s);
 for(i=0;i<=NoSc;i++) WFC[i].WFS(s);
```

```
for(i=1;i<=s-3;i+=2) Tiv[(i+1)/2]=Ts[i]-Ts[i+2];
for(i=1;i<=NoSh;i++)
        for(j=1;j<=s-1;j+=2)
                { if(Ts[j]<TmpH[i].input && Ts[j]>=TmpH[i].output) WFH[i].w[j]=-FCpSh[i];
                  elseWFH[i].w[j] = 0;
                }
for(i=1;i<=NoSc;i++)
        for(j=2;j<=s;j+=2)
                { if(Ts[j]<TmpC[i].output && Ts[j]>=TmpC[i].input) WFC[i].w[j]= FCpSc[i];
                  elseWFC[i].w[j] = 0.0;
                }
for(i=3;i<=s-1;i+=2)
{  WCpSct=0;WCpSht=0;
        for(k=1;k<=NoSh;k++) WCpSht=WCpSht+WFH[k].w[i];
        for(k=1;k<=NoSc;k++) WCpSct=WCpSct+WFC[k].w[i+1];
        SumFCp[(i-1)/2] = WCpSht+WCpSct;
}
for(i=0;i<=NoSh;i++) delete[] WFH[i].w;
for(i=0;i<=NoSc;i++) delete[] WFC[i].w;
delete[] WFH; delete[] WFC;
}

void PreAnalysis::CalQ()
{int i,j=0,n=s/2-1;;
float a,b;
a=b=0.0;  qh[0]=qc[n+1]=NULL;  QhMin=QcMin=0.0;
```

```
for(i=1;i<=n;i++) q[i] =  SumFCp[i]*Tiv[i];
for(i=1;i<=n;i++)
        { j = n+1-i;  a = a+q[i];  b = b+q[j];  qh[i] = a; qc[j] = b;
          if(qh[i]>=QhMin)  QhMin = qh[i];
          if(qc[j]<=QcMin)  QcMin = qc[j];
        }
for(i=1;i<=n;i++)
        { NetQh[i] = QhMin-qh[i-1];  NetQc[n+1-i] = QcMin-qc[n+2-i];  }
i=1; while(NetQh[i]&&NetQc[i]) i++;
Tpinch = Ts[2*i+1];  if(QhMin==0 || QcMin==0) Tpinch = -500;
}
```

ภาคผนวก ข

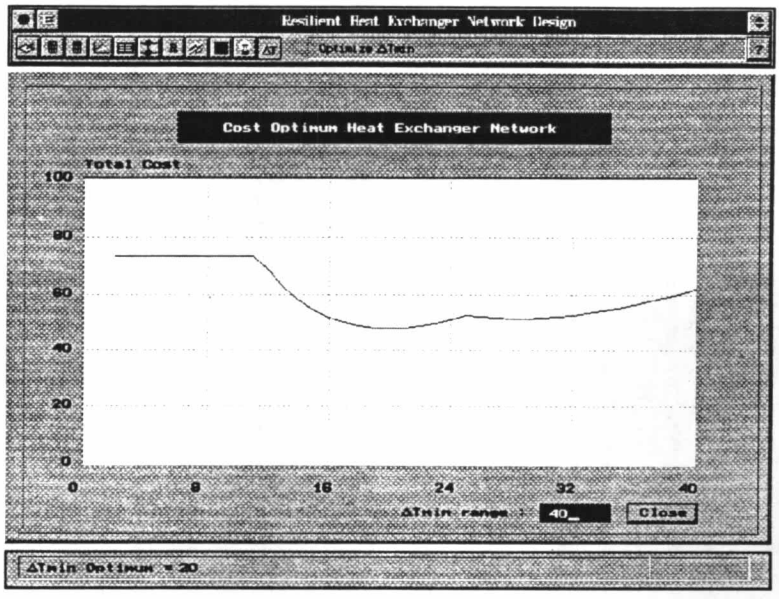
ตัวอย่างการทดสอบโปรแกรม

ตัวอย่างทดสอบโปรแกรม ข.1 ตารางข้างล่างนี้แสดงข้อมูลของกระแส สำหรับข้อมูลค่าใช้จ่ายของข่ายงานในตัวอย่างนี้และตัวอย่างต่อๆ ไปให้ใช้ข้อมูลตามตัวอย่างที่ 3.3

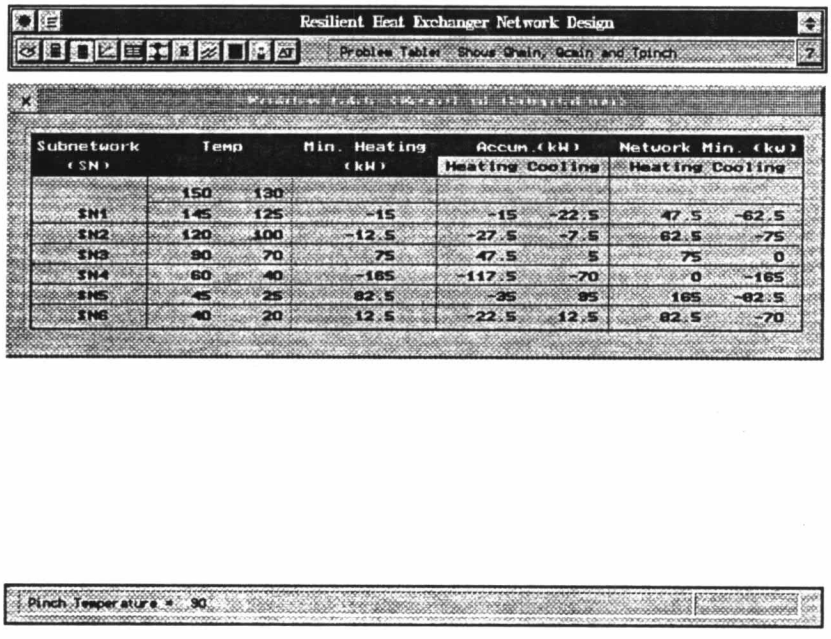
กระแส	W (kW/ $^{\circ}$C)	อุณหภูมิขาเข้า ($^{\circ}$C)	อุณหภูมิขาออก ($^{\circ}$C)	h (kW/m^{-2}.$^{\circ}$C)
H1	3.0	150	60	0.00002
H2	8.0	90	60	0.00003
C1	3.0	25	100	0.00080
C2	2.5	20	125	0.00060
HU	7.5	175	-	0.00002
CU	10.0	35	-	0.00002

ปัญหานี้ ทำการออปติไมซ์ ΔT_{min} ก่อน ซึ่งจะให้ค่า ΔT_{min} ที่ออปติมัมเท่ากับ 20 °C (ดูรูปที่ ข.1) จากการวิเคราะห์ด้วยวิธีตารางปัญหาจะได้อุณหภูมิพินช์เท่ากับ 90 °C, $Q_{H,min}$ เท่ากับ 47.5 kW และ $Q_{C,min}$ เท่ากับ 70 kW (รูปที่ ข.3) จากนั้นดูการแบ่งข่ายงานออกเป็นข่ายงานย่อยสองข่ายงานด้วยอุณหภูมิพินช์ในรูปที่ ข.4 สำหรับรูปที่ ข.5-ข.7 แสดงคำตอบของข่ายงาน

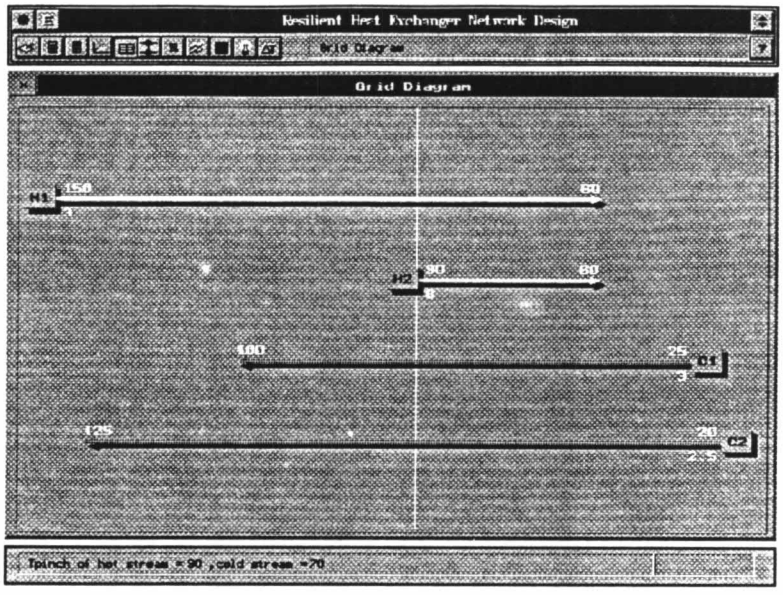
เหนือจุดพินช์ซึ่งมีทั้งหมดสามคำตอบ และรูปที่ ข.8-ข.21 แสดงคำตอบของข่ายงานใต้จุดพินช์ มีทั้งหมด 14 คำตอบ ปัญหานี้สำหรับการทดสอบการแยกกระแส (Stream splitting)



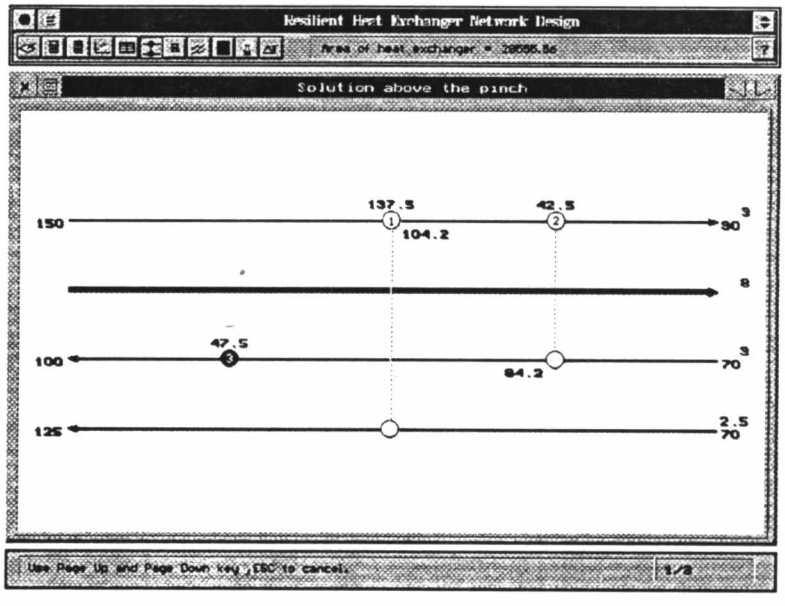
รูปที่ ข.1 แสดง ΔT_{min} ที่ออปติมัมเท่ากับ 20 °C



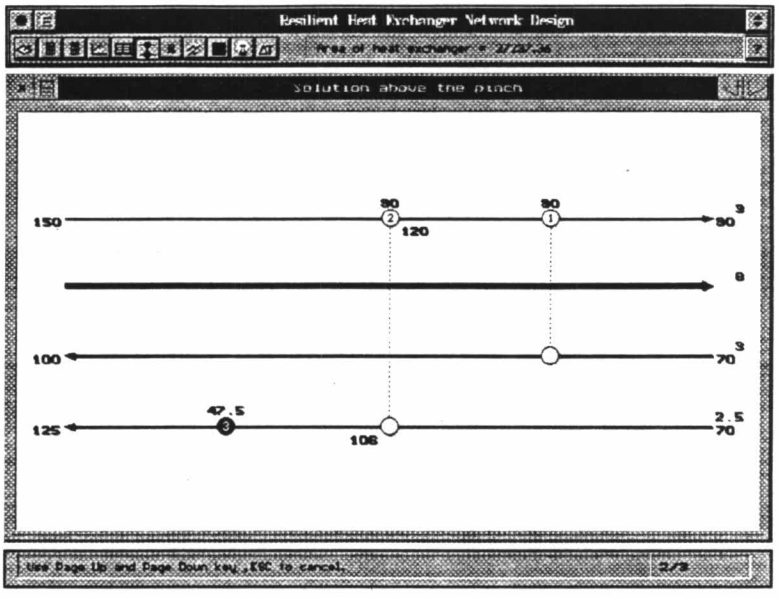
รูปที่ ข.2 ตารางปัญหา



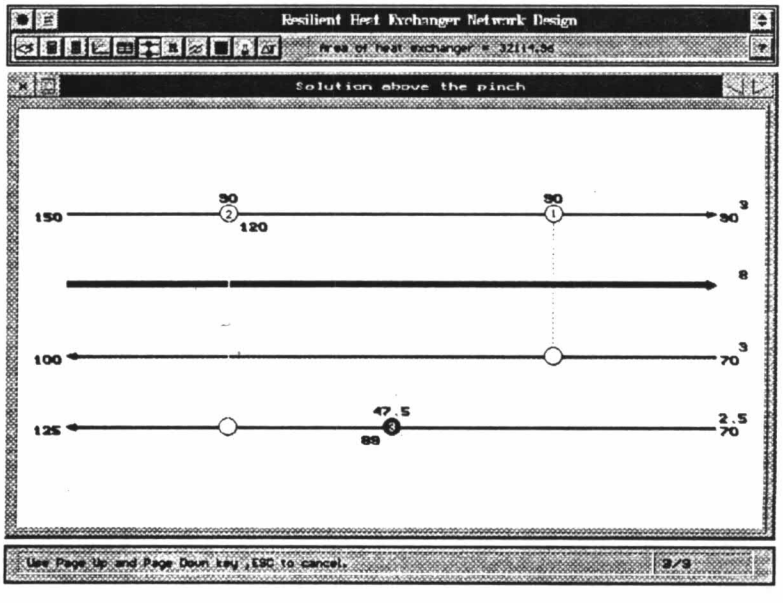
รูปที่ ข.3 Grid diagram ของตัวอย่างที่ ข.1.



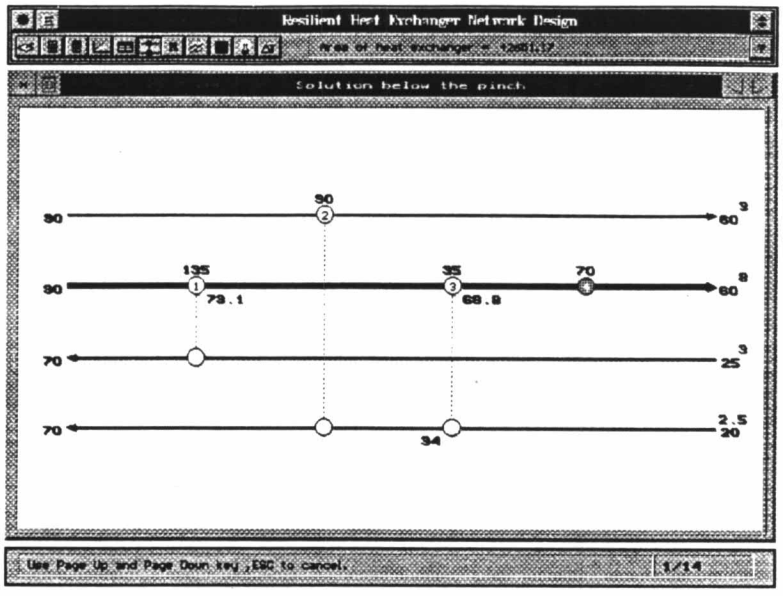
รูปที่ ข.4 ข่ายงานเหนือจุดพินช์คำตอบที่ 1/3.



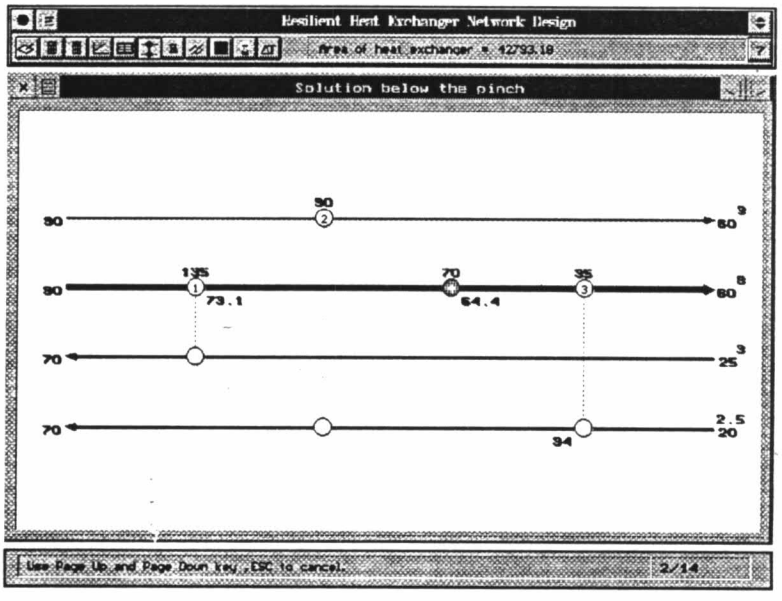
รูปที่ ข.5 ข่ายงานเหนือจุดพินช์คำตอบที่ 2/3.



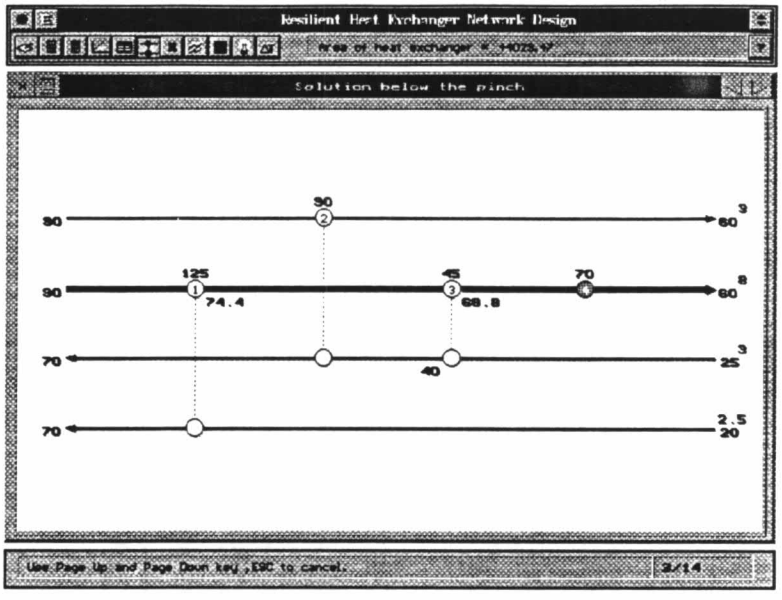
รูปที่ ข.6 ข่ายงานเหนือจุดพินช์คำตอบที่ 3/3.



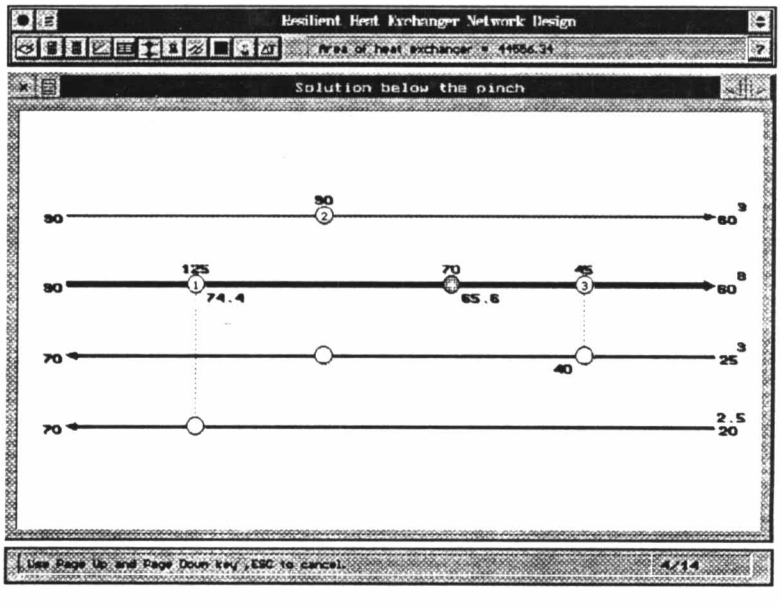
รูปที่ ข.7 ข่ายงานใต้จุดพินช์คำตอบที่ 1/14.



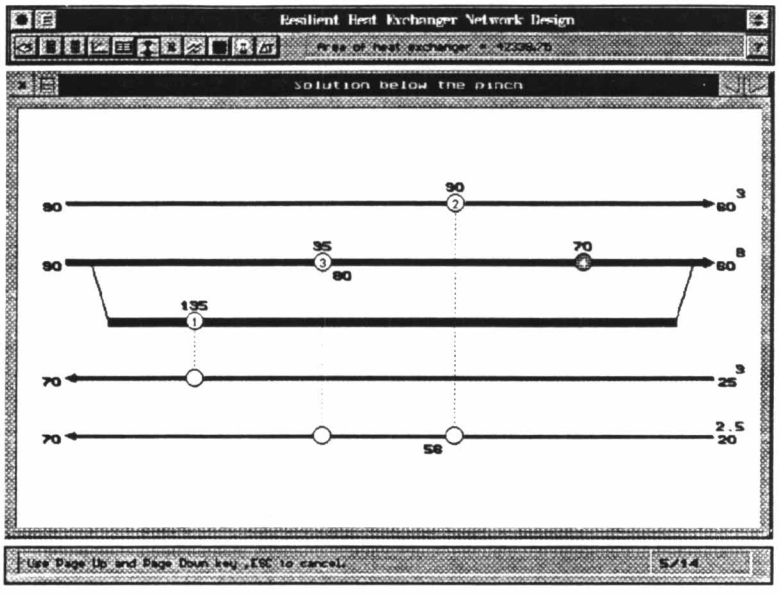
รูปที่ ข.8 ข่ายงานใต้จุดพินช์คำตอบที่ 2/14.



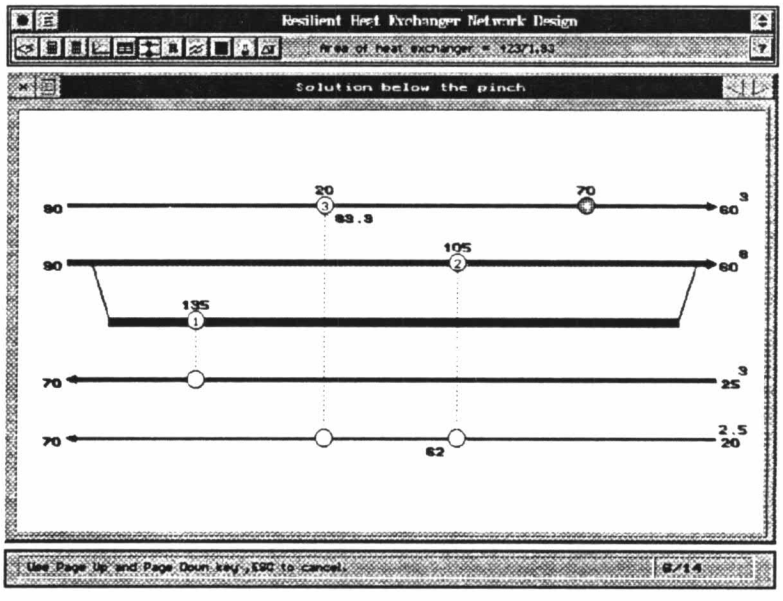
รูปที่ ข.9 ข่ายงานใต้จุดพินช์คำตอบที่ 3/14.



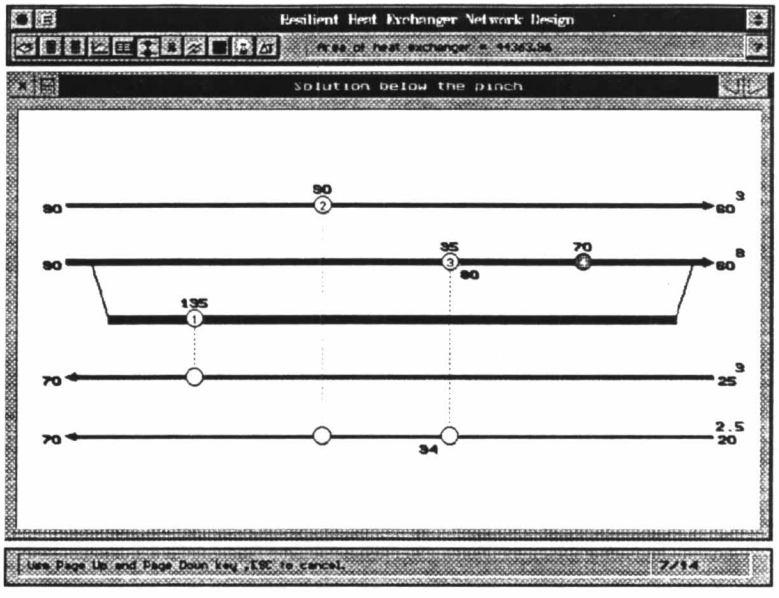
รูปที่ ข.10 ข่ายงานใต้จุดพินช์คำตอบที่ 4/14.



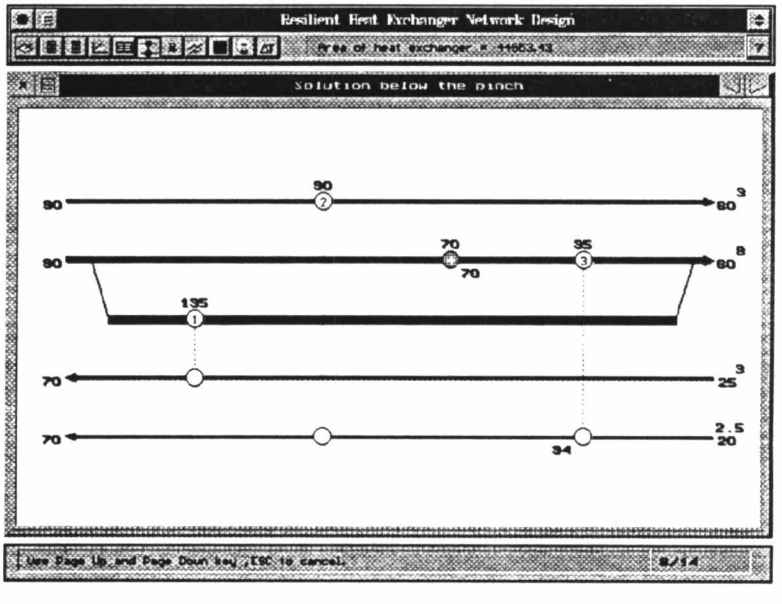
รูปที่ ข.11 ข่ายงานใต้จุดพินช์คำตอบที่ 5/14.



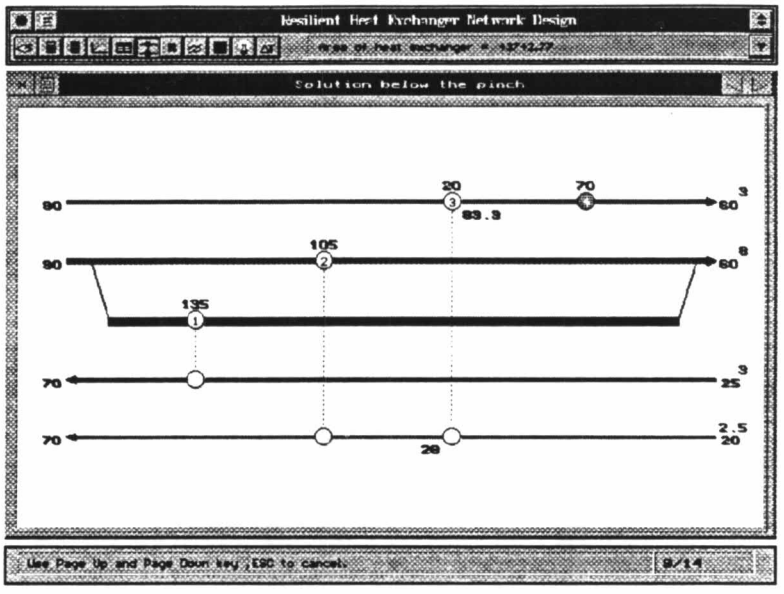
รูปที่ ข.12 ข่ายงานใต้จุดพินช์คำตอบที่ 6/14.



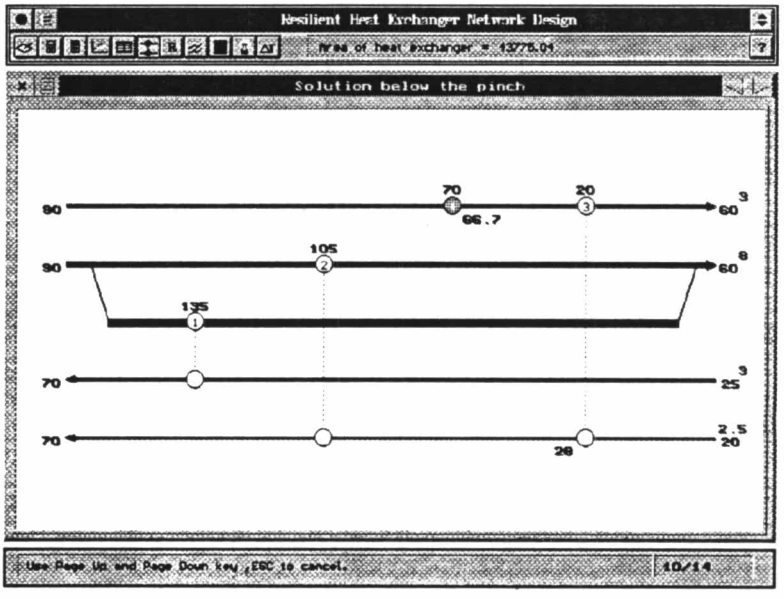
รูปที่ ข.13 ข่ายงานใต้จุดพินช์คำตอบที่ 7/14.



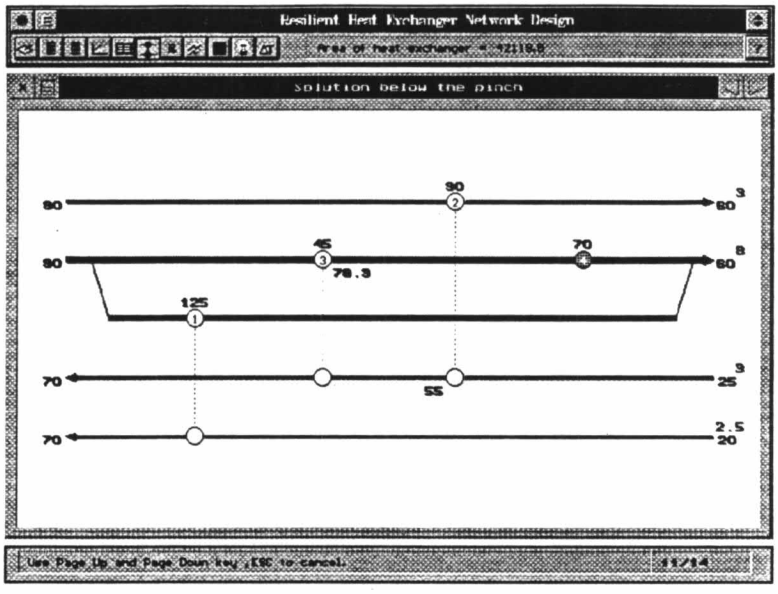
รูปที่ ข.14 ข่ายงานใต้จุดพินช์คำตอบที่ 8/14.



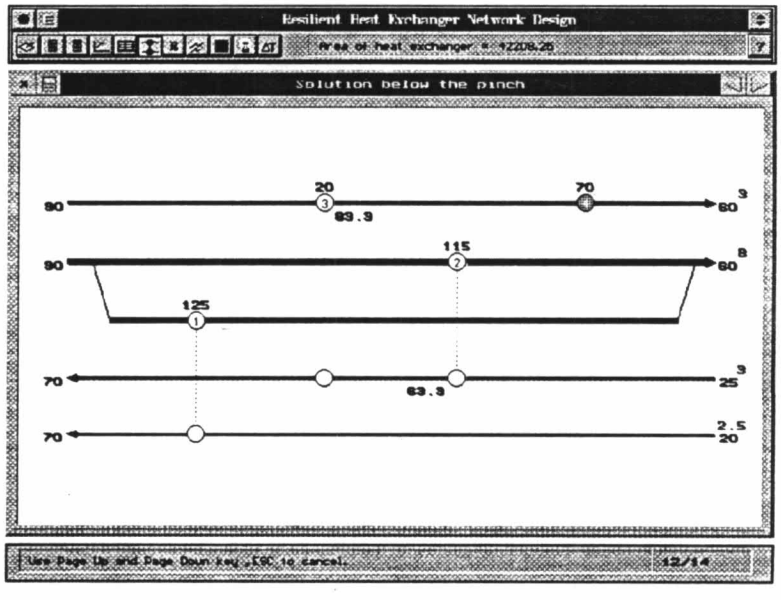
รูปที่ ข.15 ข่ายงานใต้จุดพินช์คำตอบที่ 9/14.



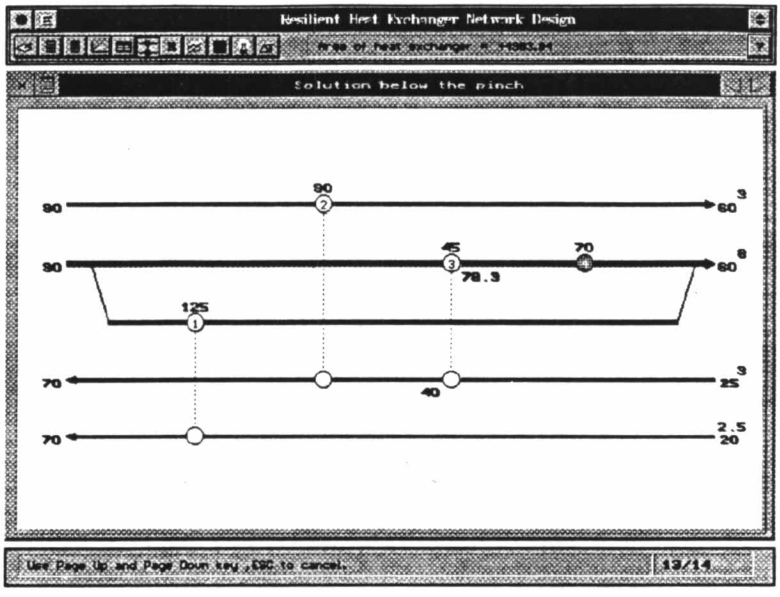
รูปที่ ข.16 ข่ายงานใต้จุดพินช์คำตอบที่ 10/14.



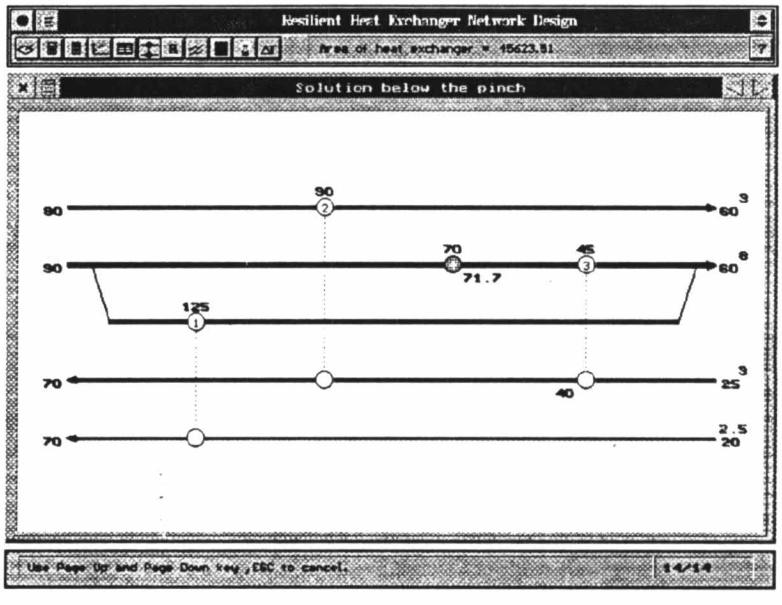
รูปที่ ข.17 ข่ายงานใต้จุดพินช์คำตอบที่ 11/14.



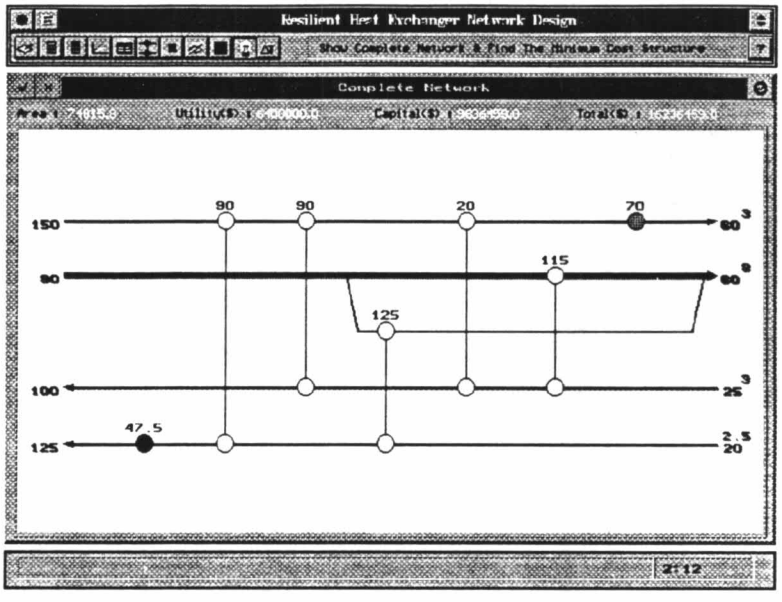
รูปที่ ข.18 ข่ายงานใต้จุดพินช์คำตอบที่ 12/14.



รูปที่ ข.19 ข่ายงานใต้จุดพินช์คำตอบที่ 13/14.



รูปที่ ข.20 ข่ายงานใต้จุดพินช์คำตอบที่ 14/14.

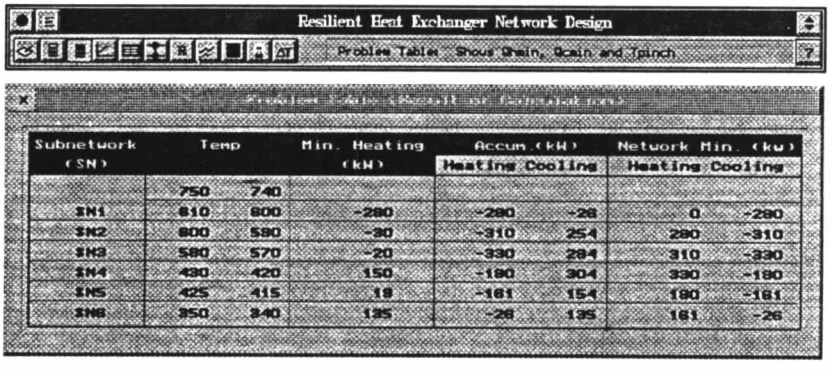


รูปที่ ข.21 แสดงข่ายงานที่สมบูรณ์ โดยรวมข่ายงานเหนือจุดพินช์และใต้จุดพินช์เข้าด้วยกัน ในที่นี้ให้โปรแกรมทำการเลือกโครงสร้างที่ประหยัดค่าใช้จ่ายที่สุดด้วยแล้ว (ได้โครงสร้างเหนือจุดพินช์คำตอบที่ 2 และใต้จุดพินช์คำตอบที่ 12)

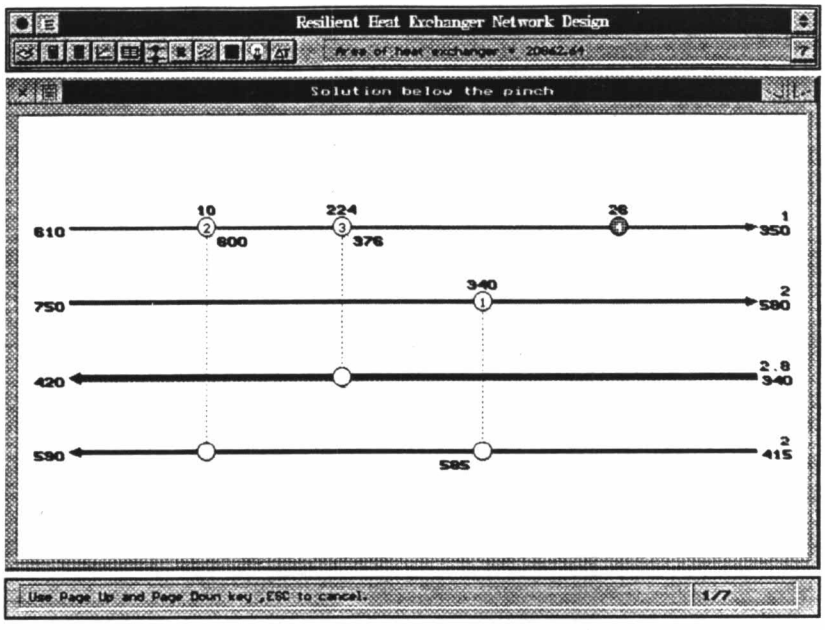
ตัวอย่างทดสอบโปรแกรม ข.2 กำหนด ΔT_{min} = 10 °C (จากตัวอย่างของบทที่ 4)

กระแส	W (kW/ °C)		อุณหภูมิขาเข้า (°C)		อุณหภูมิขาออก
	สูงสุด	ต่ำสุด	สูงสุด	ต่ำสุด	(°C)
H1	1.85	1.00	615	610	350
H2	-	2.00	-	750	580
C1	3.00	2.80	-	340	420
C2	-	2.00	-	415	590

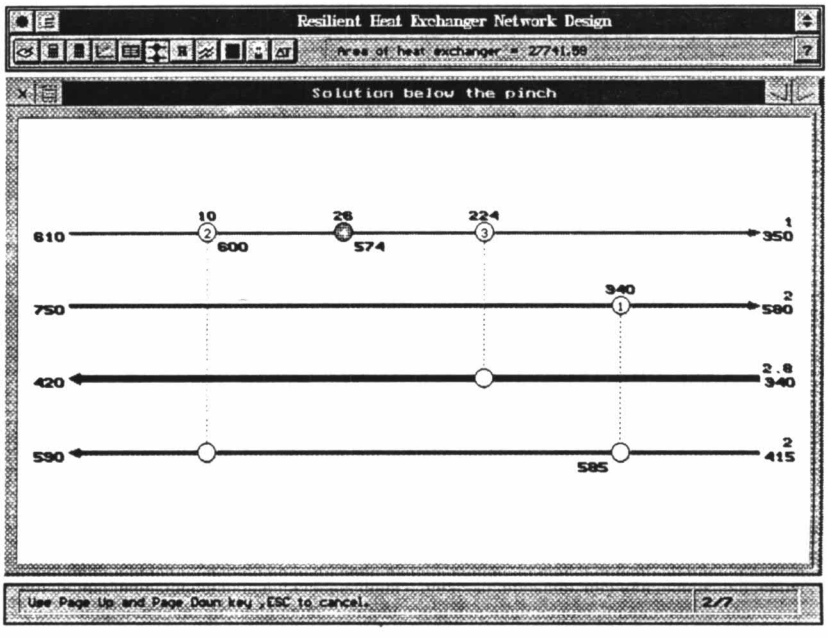
ปัญหานี้ไม่มีอุณหภูมิพินช์ ค่า $Q_{H,min}$ = 0 โดย $Q_{C,min}$ = 26 kW หรือเรียกว่าเป็นปัญหาเกี่ยวกับทางด้านความเย็นอย่างเดียว (สร้างคูลเลอร์ในข่ายงานที่กระแสร้อน ไม่มีฮีตเตอร์) สำหรับตารางปัญหาให้ดูรูปที่ ข.22 และโครงสร้างข่ายงานเมื่อไม่คิดความแปรปรวนหรือเป็นข่ายงานแบบไม่ยืดหยุ่นนั้น จะได้คำตอบทั้งหมด 7 โครงสร้างด้วยกัน (รูปที่ ข.23-ข.29) และเมื่อใช้เงื่อนไขข่ายงานแบบยืดหยุ่นแล้วจะได้โครงสร้างเพียง 1 โครงสร้างเท่านั้น (รูปที่ ข.30)



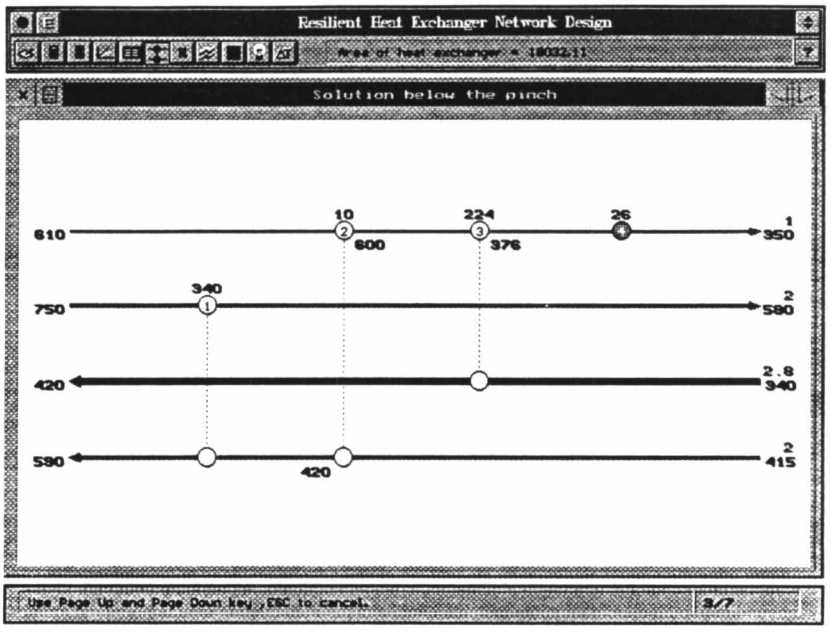
รูปที่ ข.22 แสดงตารางปัญหา



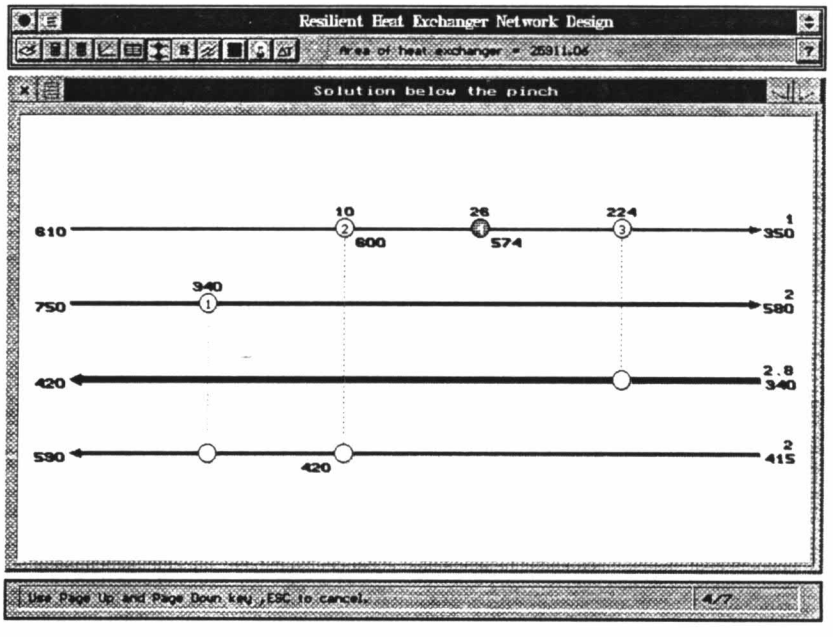
รูปที่ ข.23 ข่ายงานคำตอบที่ 1/7



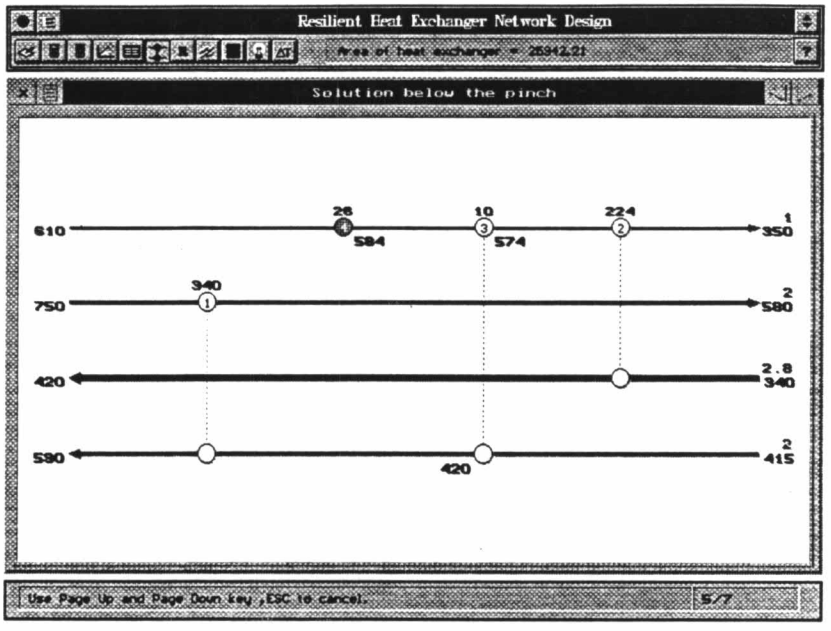
รูปที่ ข.24 ข่ายงานคำตอบที่ 2/7



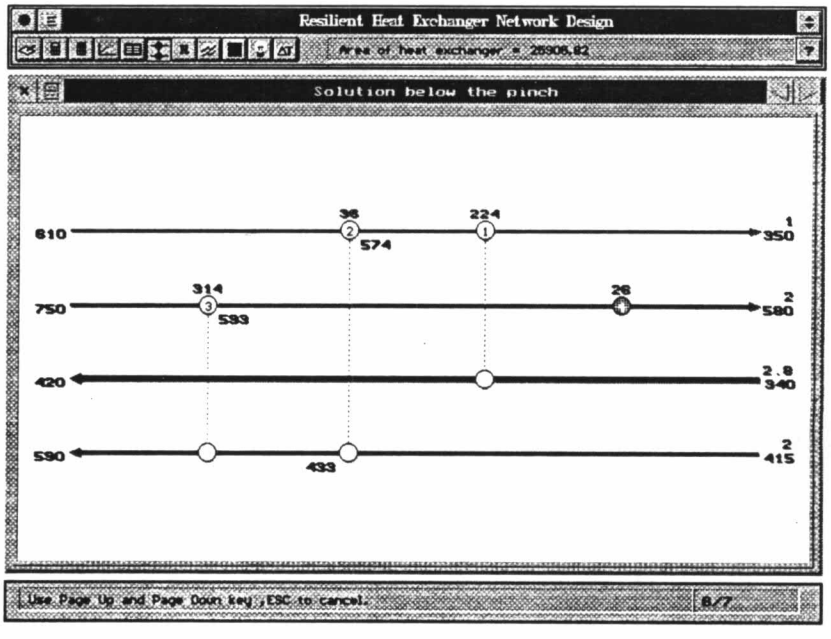
รูปที่ ข.25 ข่ายงานคำตอบที่ 3/7



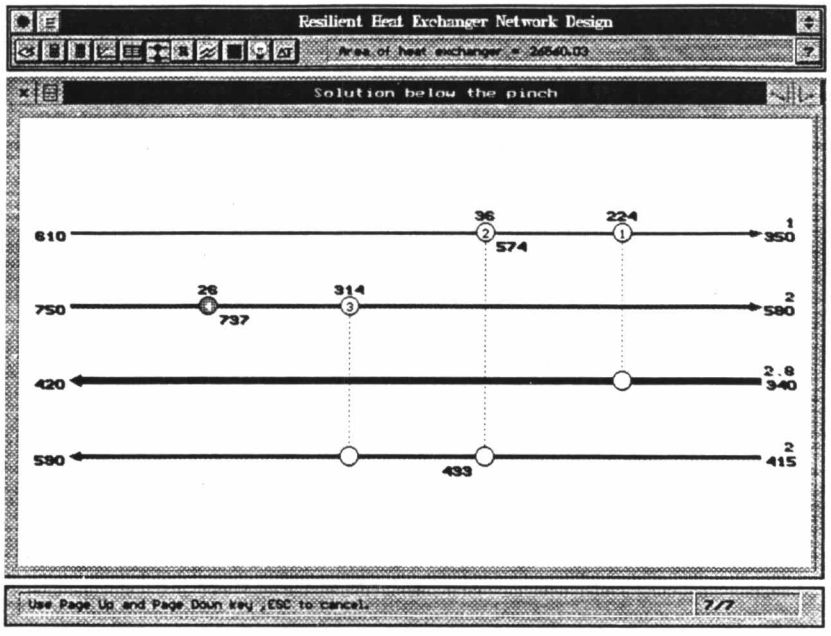
รูปที่ ข.26 ข่ายงานคำตอบที่ 4/7



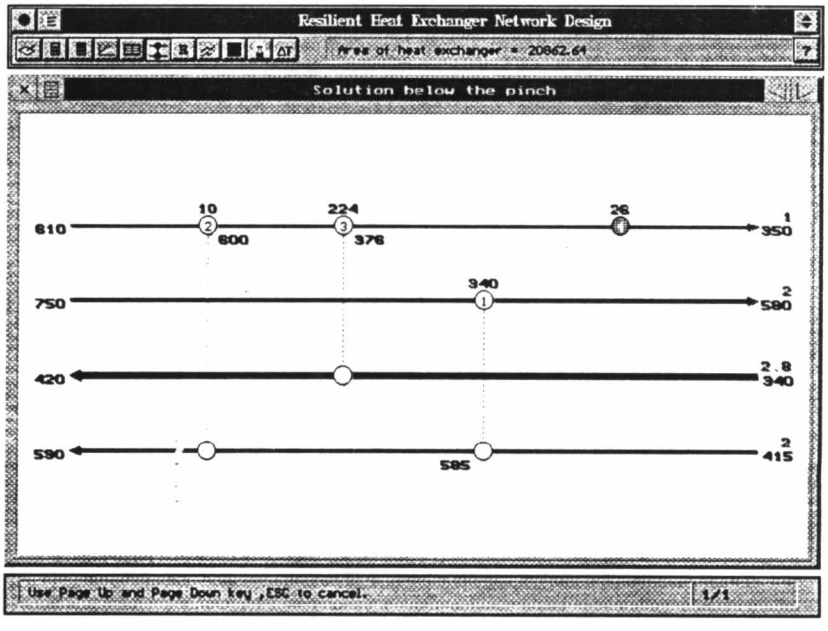
รูปที่ ข.27 ข่ายงานคำตอบที่ 5/7



รูปที่ ข.28 ข่ายงานคำตอบที่ 6/7



รูปที่ ข.29 ข่ายงานคำตอบที่ 7/7

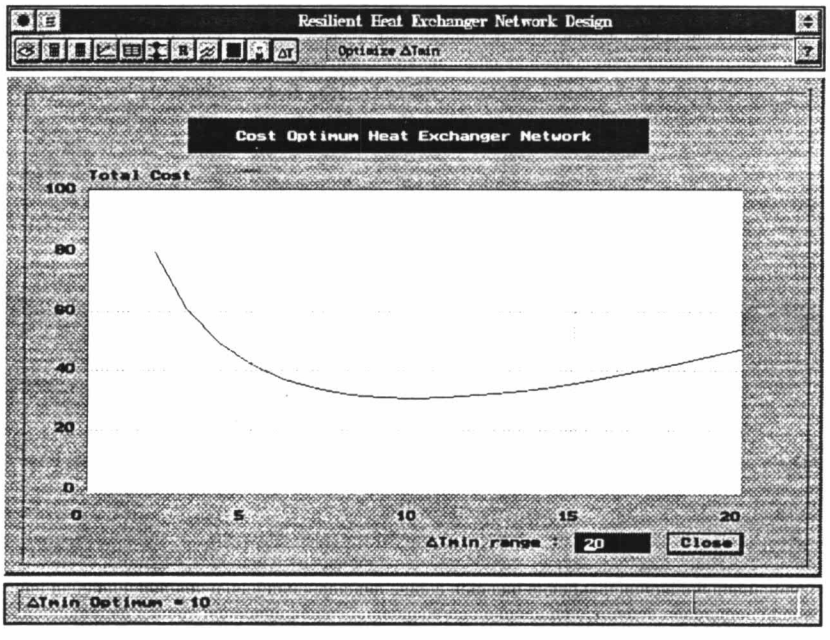


รูปที่ ข.30 แสดงข่ายงานแบบยืดหยุ่นในที่นี้ได้เพียงโครงสร้างเดียว

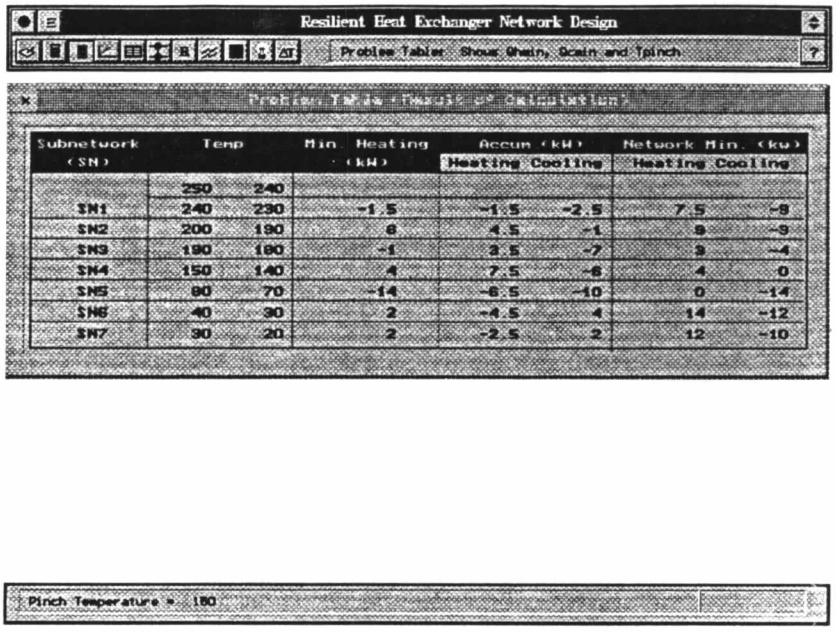
ตัวอย่างทดสอบโปรแกรม ข.3 (จากตัวอย่างที่ 3.3)

กระแส	W (kW/ °C)	อุณหภูมิขาเข้า (°C)	อุณหภูมิขาออก (°C)	h (kW/m^{-2}.°C)
H1	0.15	250	40	0.0010
H2	0.25	200	80	0.0008
C1	0.20	20	180	0.0006
C2	0.30	140	230	0.0008
HU	7.5	240	-	0.0030
CU	1.0	20	-	0.0010

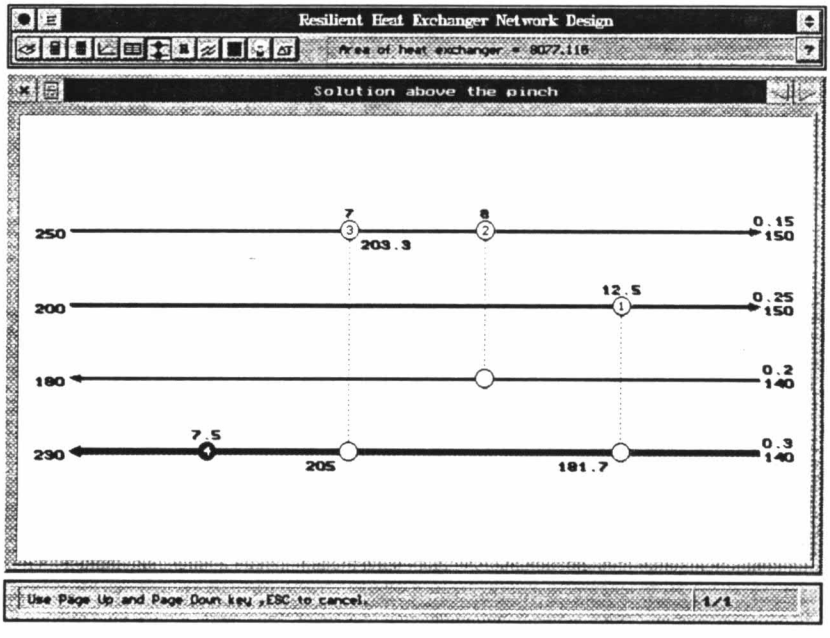
รูปที่ ข.31 แสดงการออปติไมซ์ ΔT_{min} ในเส้นโค้งคอมโพสิตสมดุล ซึ่งจะได้ค่าออกมาที่ 10 °C จากวิธีตารางปัญหา (รูปที่ ข.32) ได้ $Q_{H,min}$ เท่ากับ 7.5 kW, $Q_{C,min}$ เท่ากับ 10 kW และอุณหภูมิพินช์เท่ากับ 150 °C รูปที่ ข.33-ข.36 แสดงข่ายงานเหนือจุดพินช์และใต้จุดพินช์ที่หาได้ รูปที่ ข.37 แสดงข่ายงานที่ประหยัดค่าใช้จ่ายมากที่สุดเหนือจุดพินช์และใต้จุดพินช์เข้าด้วยกัน



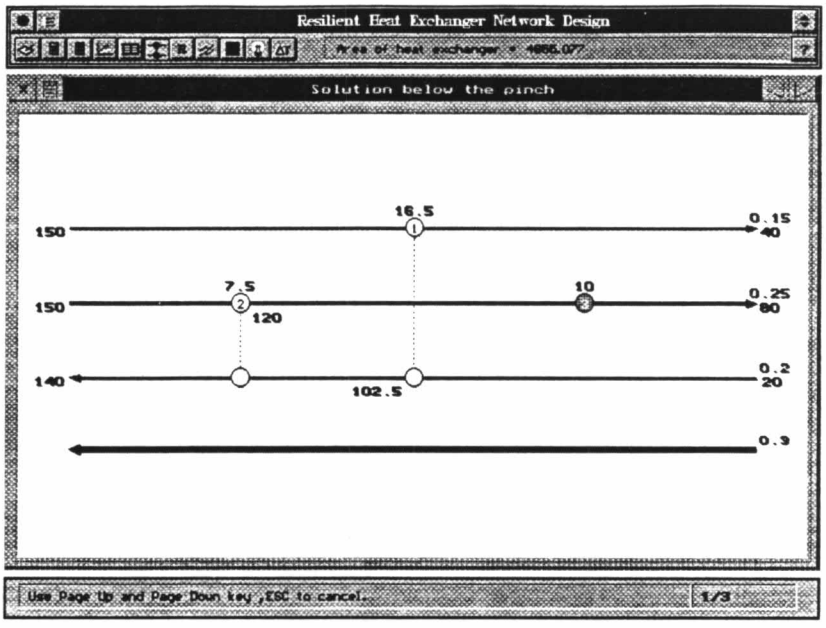
รูปที่ ข.31 ได้ค่า ΔT_{min} ที่ออปติมัมเท่ากับ 10 °C



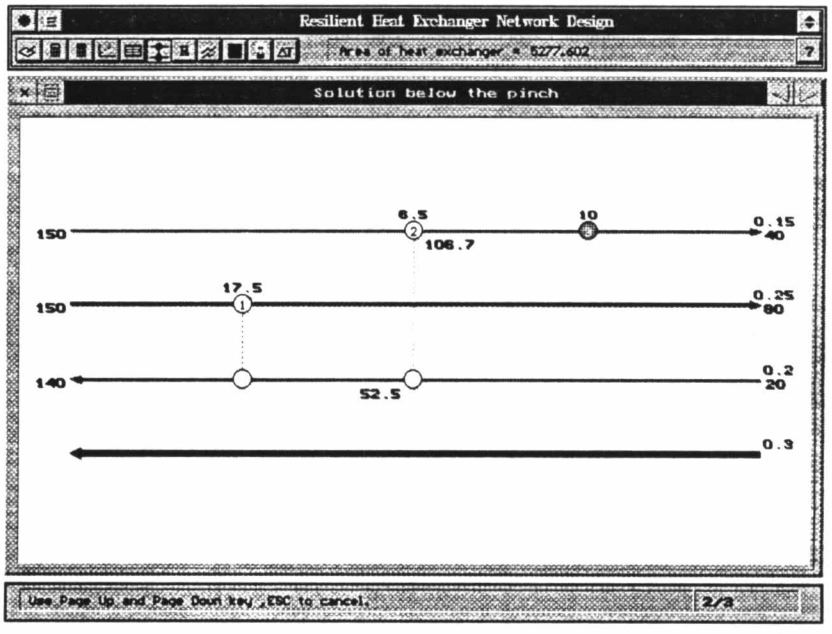
รูปที่ ข.32 ตารางปัญหา



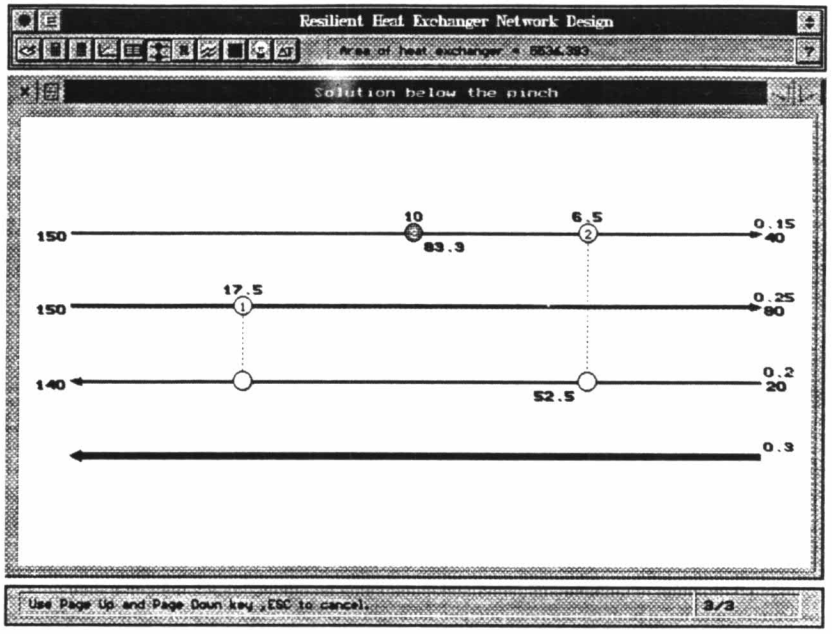
รูปที่ ข.33 แสดงข่ายงานเหนือจุดพินช์



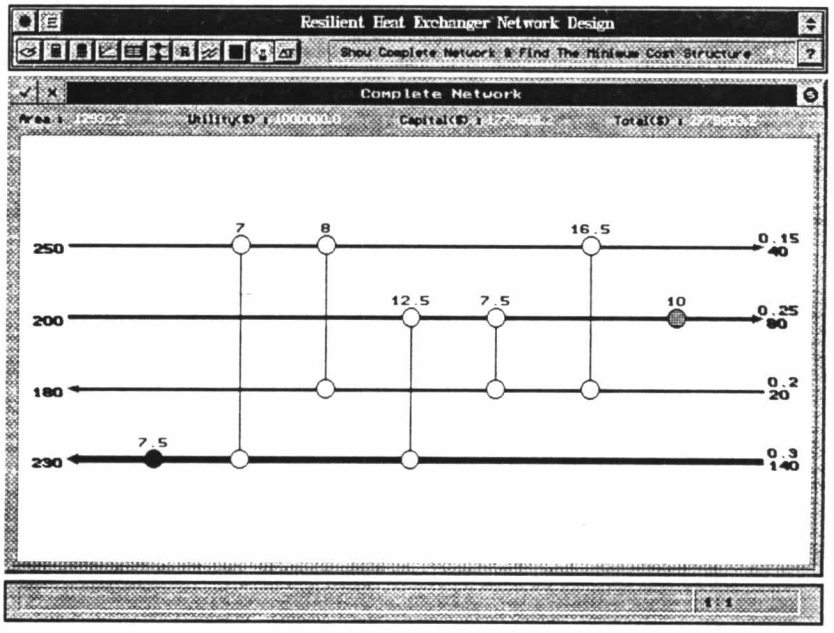
รูปที่ ข.34 แสดงข่ายงานใต้จุดพินช์คำตอบที่ 1/3



รูปที่ ข.35 แสดงข่ายงานใต้จุดพินช์คำตอบที่ 2/3



รูปที่ ข.36 แสดงข่ายงานใต้จุดพินช์คำตอบที่ 3/3



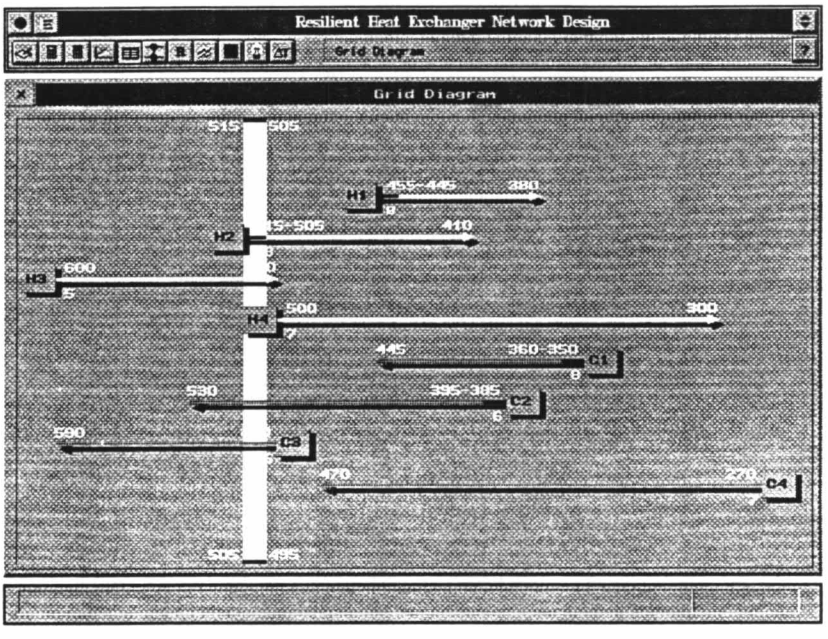
รูปที่ ข.37 แสดงข่ายงานที่สมบูรณ์และประหยัดค่าใช้จ่ายมากที่สุด

ตัวอย่างทดสอบโปรแกรม ข.4 แสดงการออกแบบข่ายงานแบบยืดหยุ่น เมื่อกระแสมีความแปรปรวนของอุณหภูมิ กำหนด ΔT_{min} = 10 °C

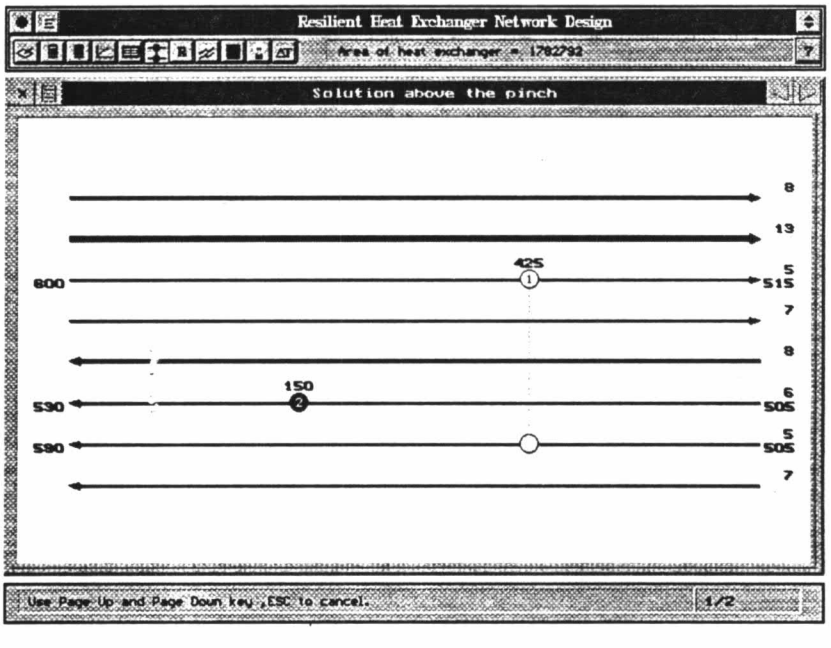
กระแส	W	อุณหภูมิขาเข้า (°C)			อุณหภูมิขาออก
	(kW/°C)	ปกติ	สูงสุด	ต่ำสุด	(°C)
H1	8.0	450	455	445	380
H2	13.0	510	515	505	410
H3	5.0	600	600	600	500
H4	7.0	500	500	500	300
C1	8.0	355	360	350	445
C2	6.0	390	395	385	530
C3	5.0	490	490	490	590
C4	7.0	270	270	270	470
HU	7.5	600			
CU	10.0	200			

ในปัญหานี้ มีกระแสกระบวนการร้อนและเย็นรวมทั้งสิ้น 8 กระแส ความแปรปรวนของอุณหภูมิขาเข้าของกระแส H1, H2, C1 และ C2 อยู่ในช่วง ±5 °C จากภาวะปกติ โดยใช้วิธีตารางปัญหาพบว่า ข่ายงานมีความแปรปรวนของอุณหภูมิพินช์ในช่วง 505-515 °C รูปที่ ข.38 แสดงกริดไดอะแกรมของกระแสและอุณหภูมิพินช์ โดยการเลือกโหมดการออกแบบข่ายงานแบบยืดหยุ่น จะได้คำตอบของข่ายงานเหนือจุดพินช์ 2 คำตอบ และคำตอบของข่ายงานใต้จุดพินช์ 1 คำตอบ (รูปที่ 39-41) และรูปที่ 42 แสดงการรวมข่ายงานย่อยทั้งสองเข้าด้วยกันโดยโปรแกรมเลือกข่ายงานเหนือจุดพินช์คำตอบที่ 2 เพราะว่าจะประหยัดค่าใช้จ่ายของข่ายงานมาก

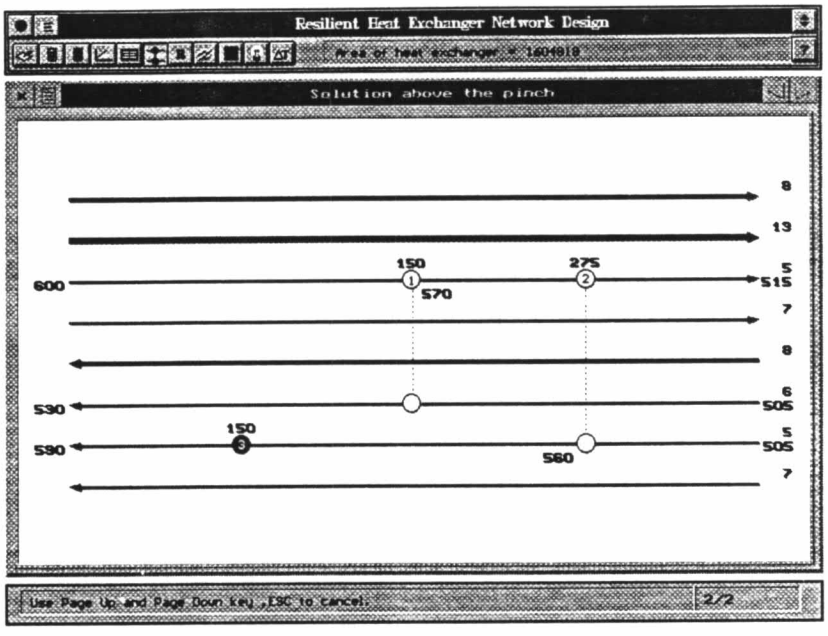
กว่าคำตอบที่ 1 สำหรับรูปที่ 43-60 แสดงคำตอบของการออกแบบข่ายงานเหนือจุดพินช์และใต้จุดพินช์แบบ ไม่ยืดหยุ่น



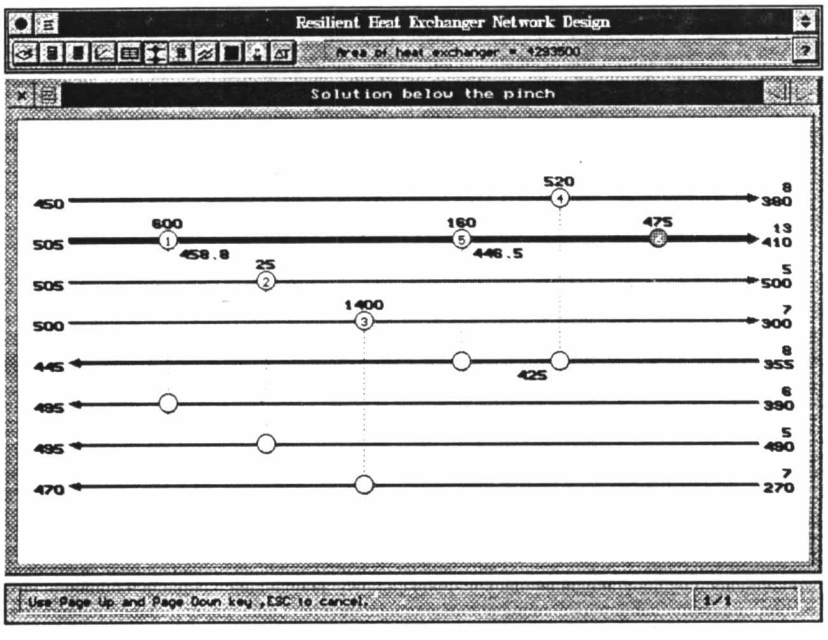
รูปที่ ข.38 กริดไดอะแกรมแสดงช่วงความแปรปรวนของอุณภูมิพินช์



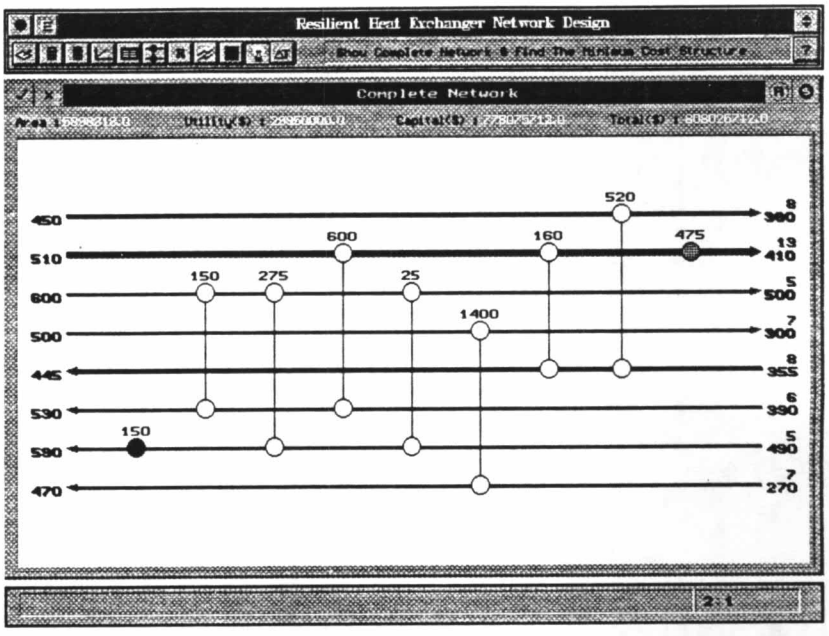
รูปที่ ข.39 ข่ายงานแบบยืดหยุ่นเหนือจุดพินช์คำตอบที่ 1/2



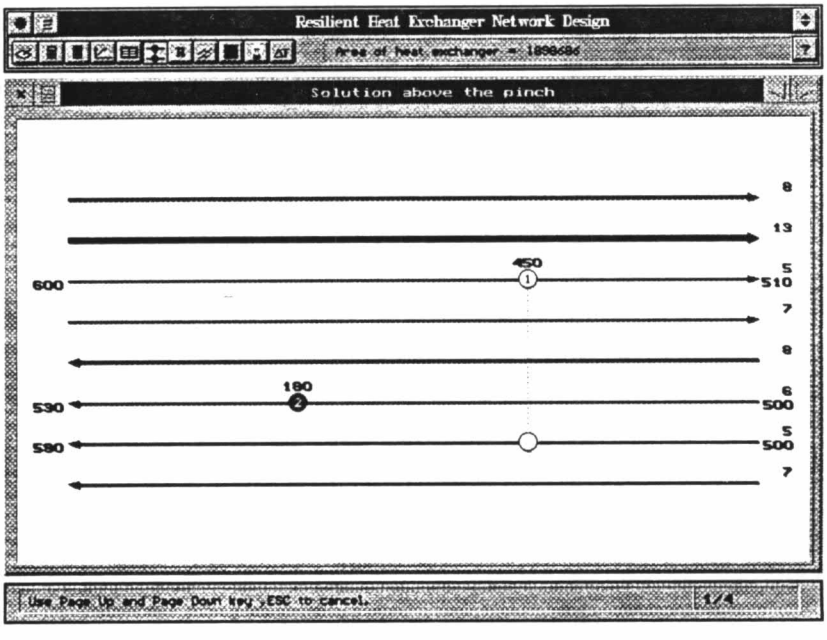
รูปที่ ข.40 ข่ายงานแบบยืดหยุ่นเหนือจุดพินช์คำตอบที่ 2/2



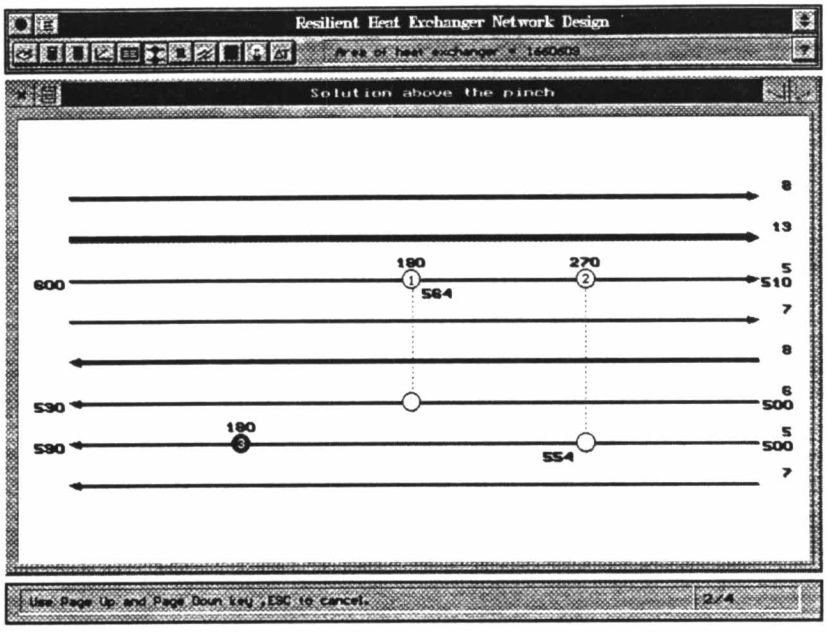
รูปที่ ข.41 ข่ายงานแบบยืดหยุ่นใต้จุดพินช์คำตอบที่ 1/1



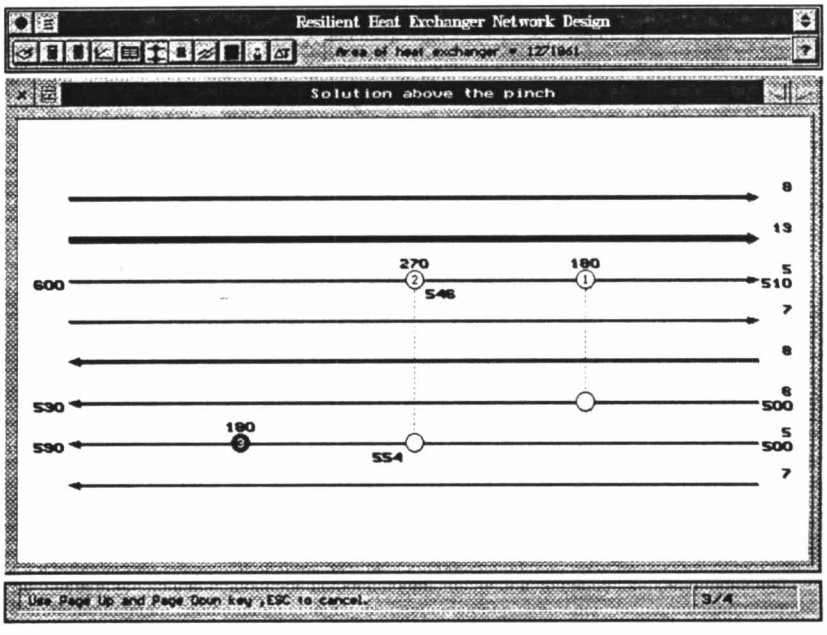
รูปที่ ข.42 แสดงข่ายงานรวม



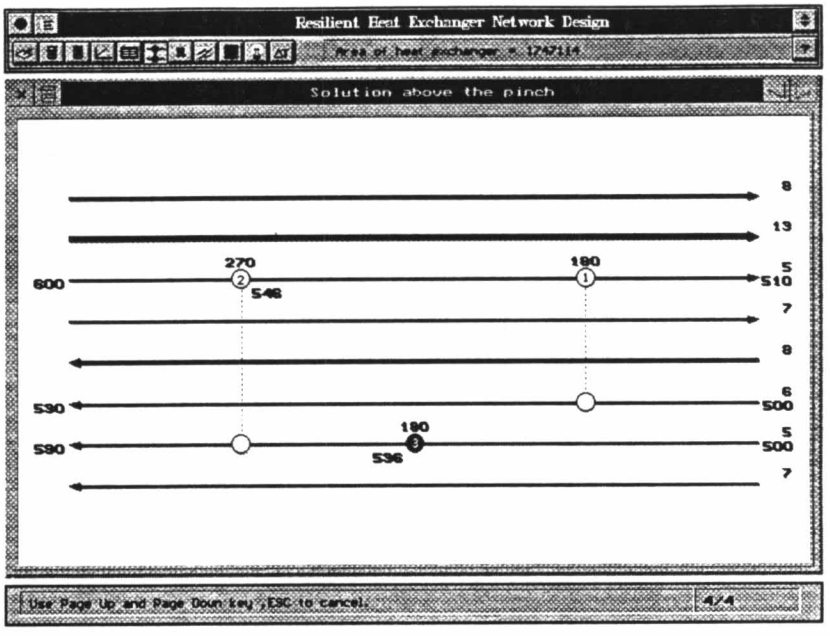
รูปที่ ข.43 ข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์คำตอบที่ 1/4



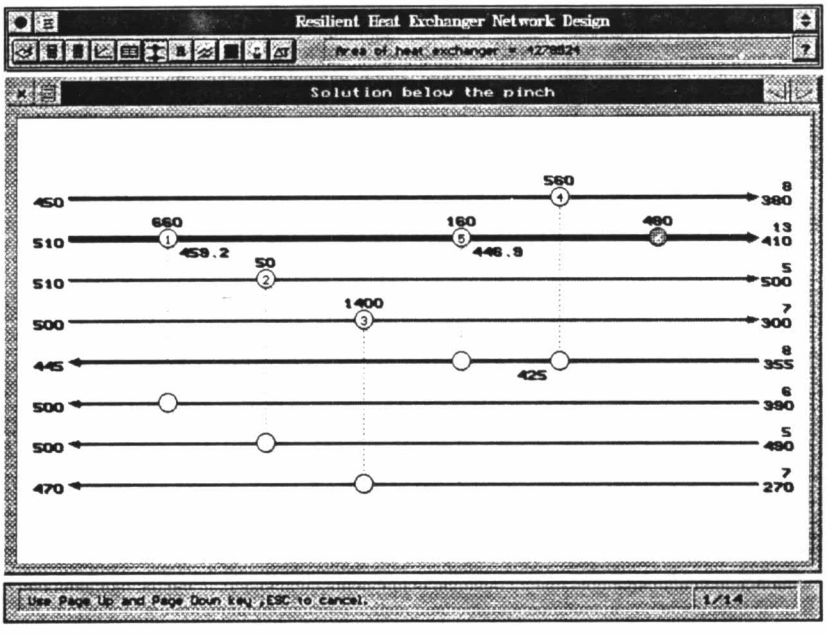
รูปที่ ข.44 ข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์คำตอบที่ 2/4



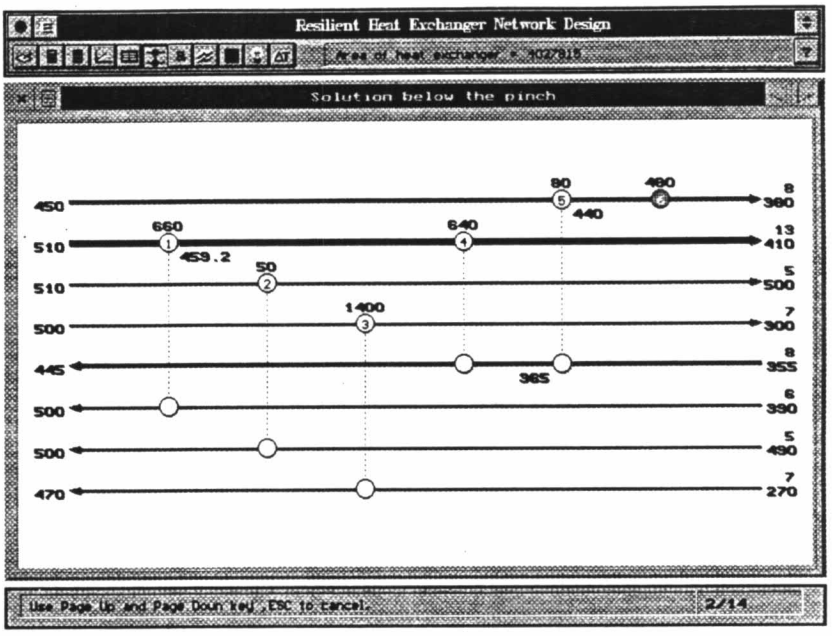
รูปที่ ข.45 ข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์คำตอบที่ 3/4



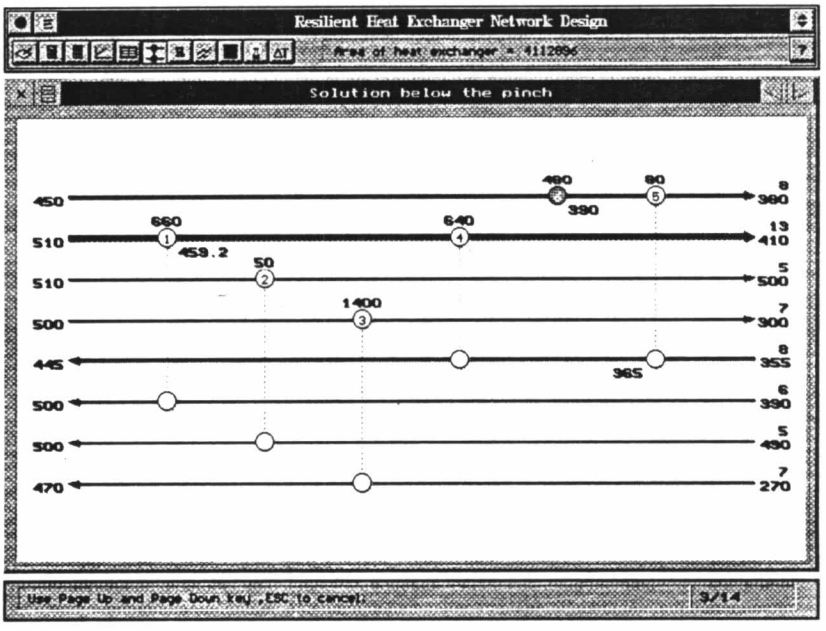
รูปที่ ข.46 ข่ายงานแบบไม่ยืดหยุ่นเหนือจุดพินช์คำตอบที่ 4/4



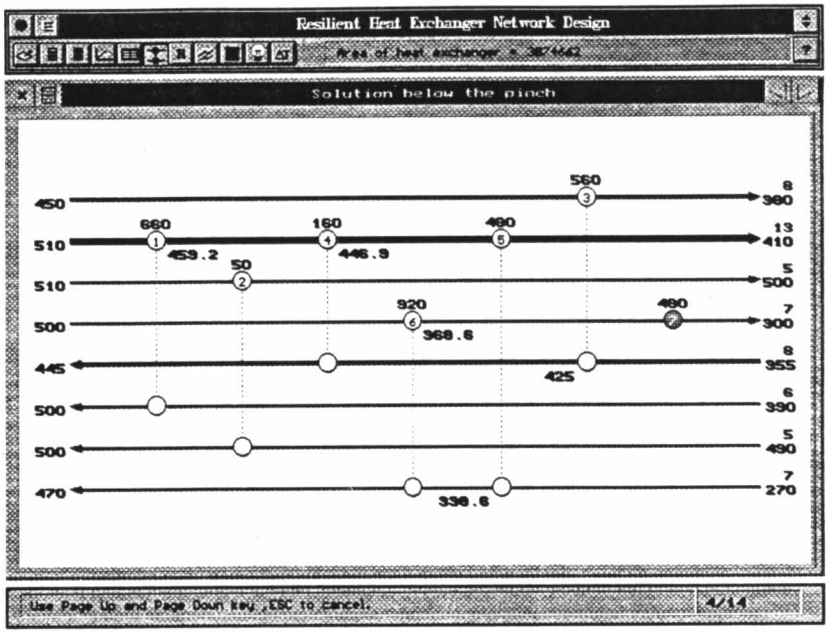
รูปที่ ข.47 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 1/14



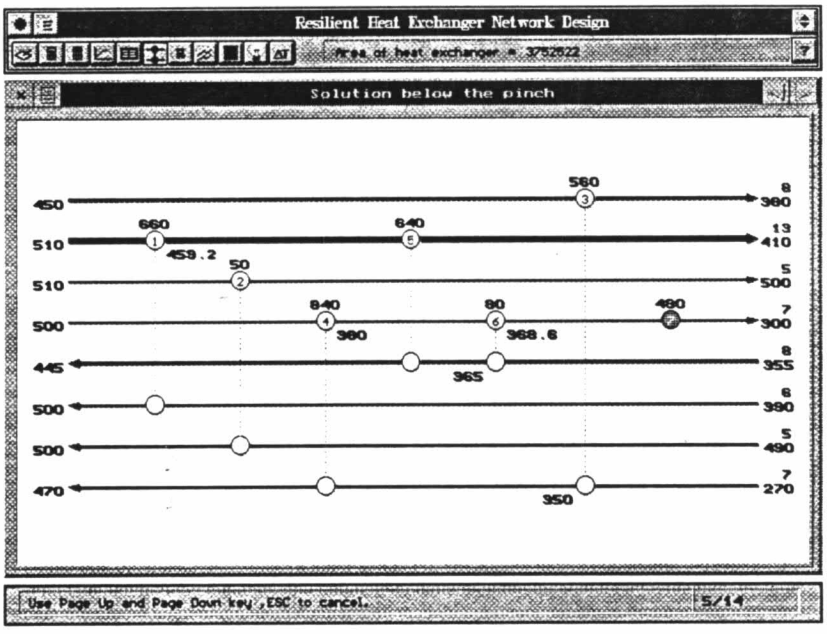
รูปที่ ข.48 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 2/14



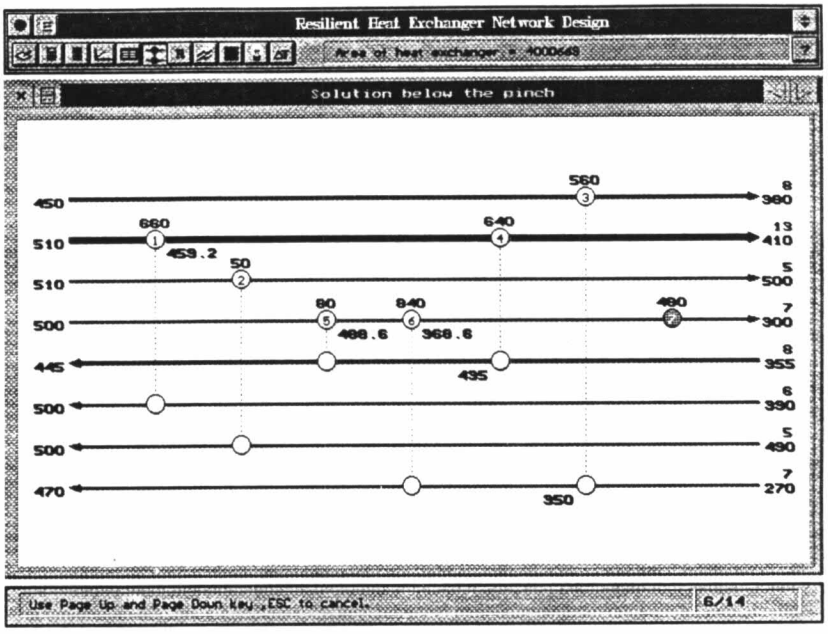
รูปที่ ข.49 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 3/14



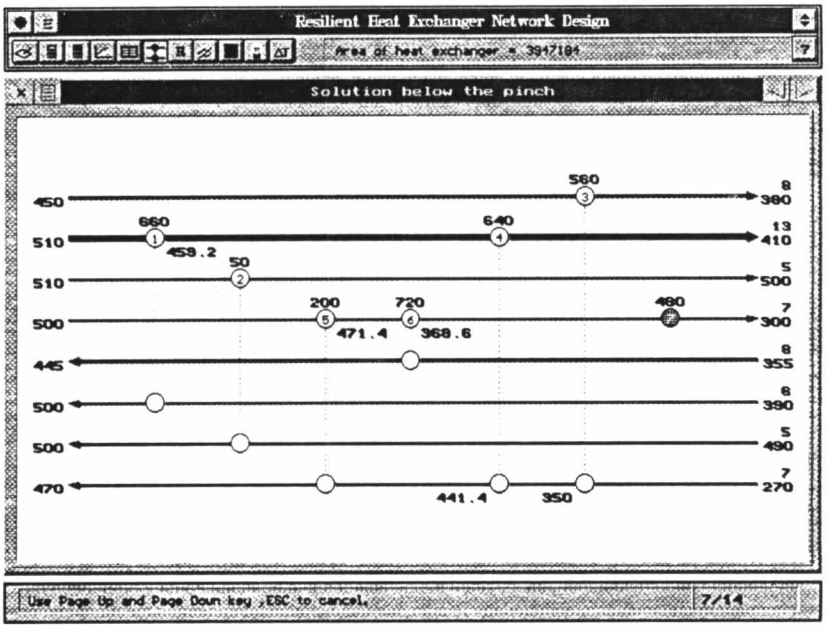
รูปที่ ข.50 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 4/14



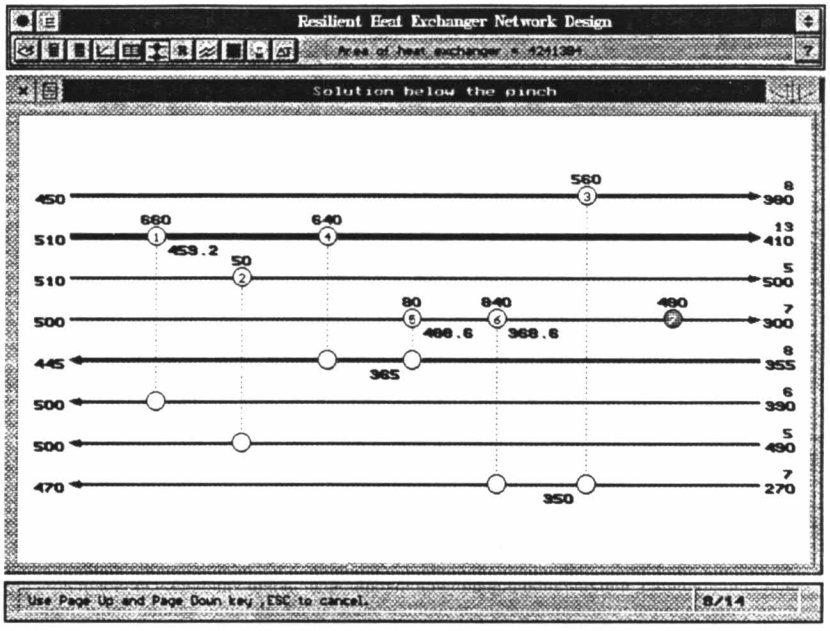
รูปที่ ข.51 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 5/14



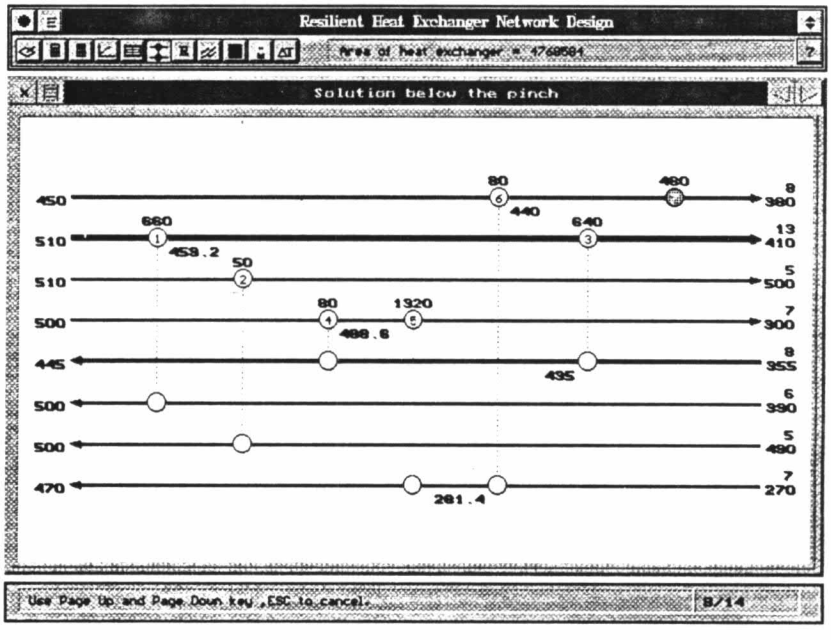
รูปที่ ข.52 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 6/14



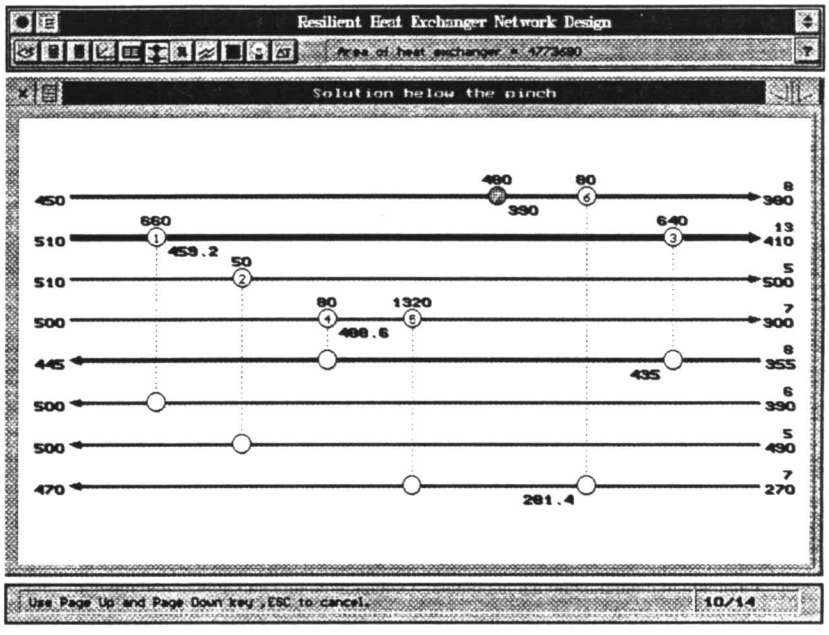
รูปที่ ข.53 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 7/14



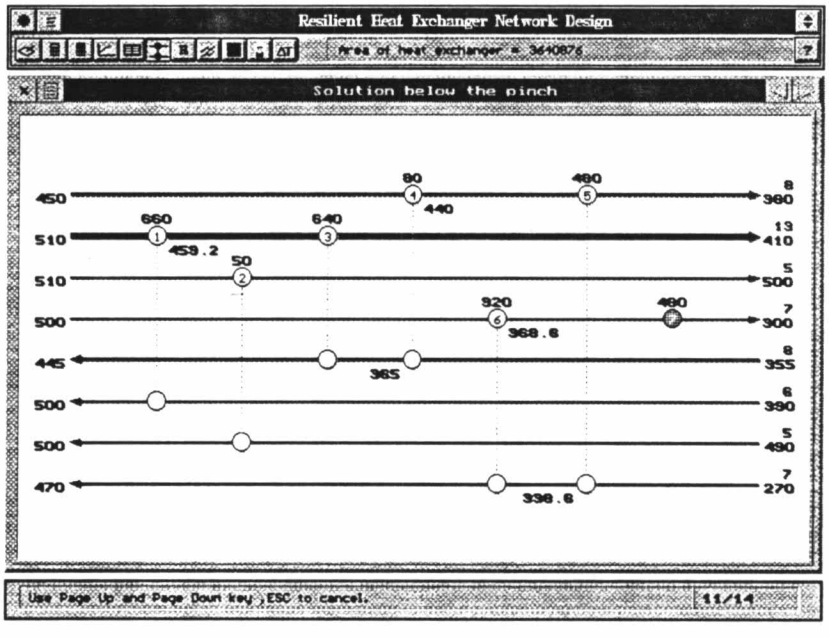
รูปที่ ข.54 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 8/14



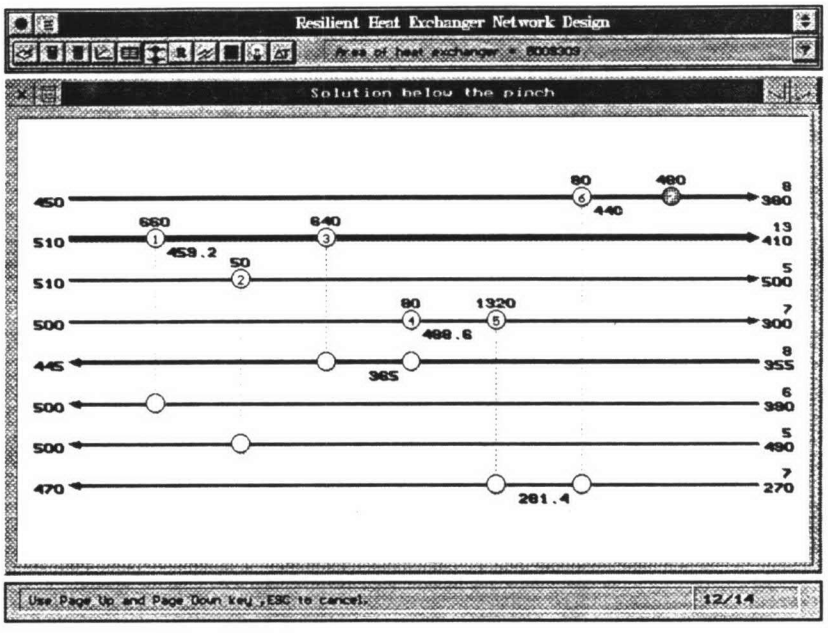
รูปที่ ข.55 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 9/14



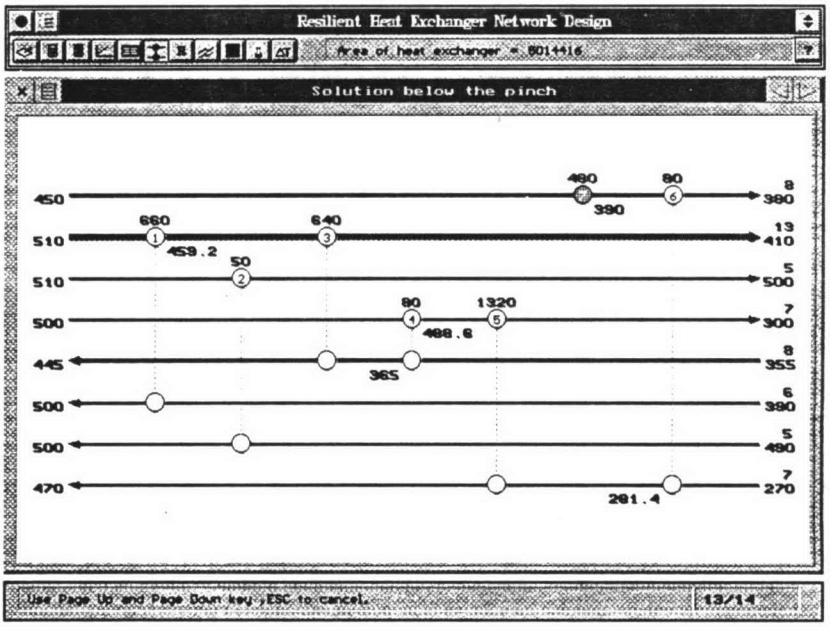
รูปที่ ข.56 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 10/14



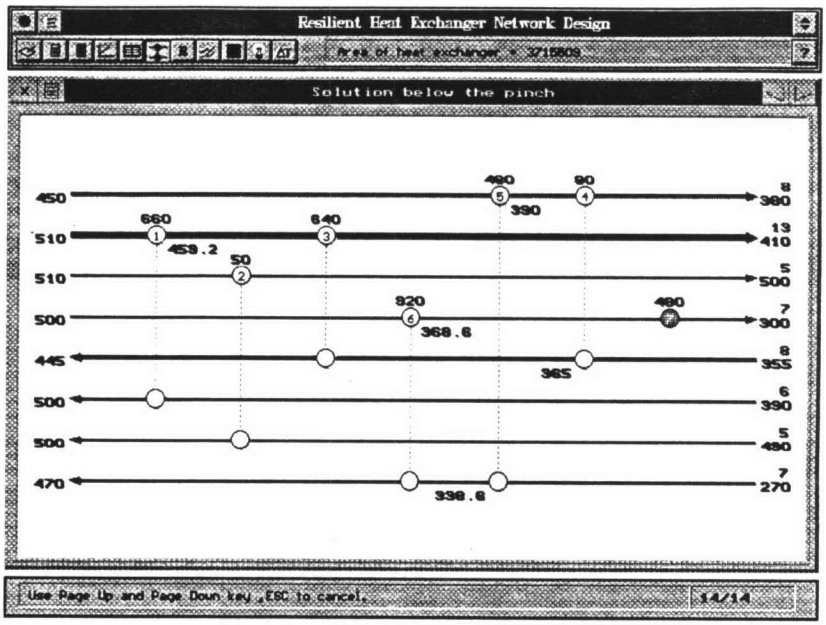
รูปที่ ข.57 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 11/14



รูปที่ ข.58 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 12/14



รูปที่ ข.59 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 13/14



รูปที่ ข.60 ข่ายงานแบบไม่ยืดหยุ่นใต้จุดพินช์คำตอบที่ 14/14

ประวัติผู้เขียน

นายณัฐพร ทรงศิริ เกิดเมื่อวันที่ 26 พฤษภาคม 2514 สำเร็จการศึกษาในระดับมัธยมศึกษาตอนปลายสายสามัญ จากโรงเรียนสาธิตวิทยาลัยครูเทพสตรี จังหวัดลพบุรี เมื่อปีการศึกษา 2530 สำเร็จการศึกษาระดับปริญญาตรี (วิทยาศาสตรบัณฑิต) สาขาเคมีเทคนิค จากจุฬาลงกรณ์มหาวิทยาลัย ปีการศึกษา 2534