CHAPTER 5

COMPUTER PROGRAM FOR HEAT EXCHANGER NETWORK DESIGN USING MATCH PATTERN APPROACH

5.1 Program Overview

The HEN design program is developed on Visual Basic version 1.0 for Microsoft Windows. The program is partitioned into two major steps. The first is preanalysis and the second is network invention.

In the preanalysis step, input data is used to find network targets i.e., pinch temperature, minimum utilities requirement. In network invention step, the problem will be divided into subproblems and the subproblems will be separately solved. The design procedure is to search subnetwork design space by using match patterns to match streams together. The program will search for heat exchanger matches until the subproblems are solved. When a subnetwork solution is found, the program will try to find the others if they exist. The program will search until it can not find another solution When the program terminates, there may be several alternative solutions.

5.2 Assumption Used in The Program

1. Supply and target temperatures and heat capacity flowrates of all process streams are known and fixed.
2. No phase change occurs when streams exchange their heat loads.
3. Overall heat transfer coefficients are constant for all exchanger units.

5.3 Program Limitations

1. Maximum numbers of hot streams and cold streams are 10 streams.
2. Maximum number of units for each subnetwork is 30 units.
3. Maximum number of alternative subnetwork is 30 solutions.

5.4 Program Flow Charts

5.4.1 Preanalysis Step Function

Figure 5.1 shows how the program work in major steps. When the program starts; a user can select whether to input stream information via keyboard or by input file. The required information to design heat exchanger network are numbers of streams, supply temperatures, target temperatures, heat capacity flowrates, minimum temperature difference and overall heat transfer coefficient. All information will be stored in specific data records. Particularly, the stream information will be stored in STREAMINFO record which consists of supply and target temperatures, heat capacity flowrate, heat load and their status. The problem information will be stored in NODEINFO record.

Some of problem information will be used in the preanalysis step in which the problem table algorithm is used to determine location of the pinch point in the network. If it is a pinched problem, the split problem function will be called to split problem into two parts, a hot-end subproblem and a cold-end subproblem. The network invention function will treat each part as a subproblem depending on what utility type is required.

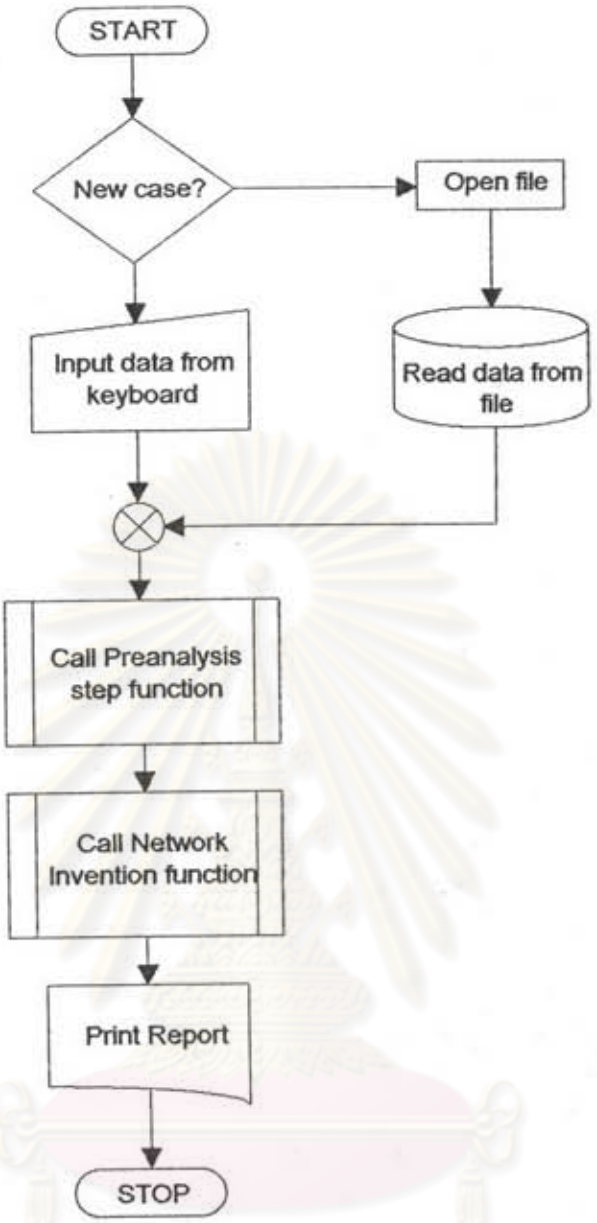

Figure 5.1 Simplified Program flow chart.

Figure 5.2 shows the flow chart of the preanalysis functions. The input information , as "ROOTNODE", will be used by the problem table to determine the pinch temperature. If it is a pinched problem, the pinch temperature will be used as a split point to divide root node to two subproblems. The first one called, hot-end subproblem and its information is stored in "HOTROOT". The second called cold-end subproblem, and its

information is stored in "COLDROOT". If the program can not find the pinch point, this means that the problem is either a heat sink or a heat source problem but not both.

5.4.2 Pinch Temperature Finder Function

The flow chart of the pinch temperature finder function is shown in Fig. 5.3. The function uses the problem table method (Linnhoff and Hindmarsh, 1983) to determine the minimum utility requirements and the location of the pinch point. The function will read supply and target temperatures of input streams from ROOTNODE and use this to set up intervals from higher to lower temperatures. Next, the function will calculate heat accumulated in each interval. If heat flow is less than zero the hot utility must be added to the highest temperature interval, TI[1]. The pinch point will lie on the temperature that heat flow equal to zero. The minimum hot utility requirement is equal to heat added to TI[1] and the minimum cold utility requirement is equal to heat flow out from the lowest temperature Interval, TI[n]. If accumulative heat is greater than zero the problem is threshold problem. The hot utility for this case is not required. The minimum cold utility requirement is equal to heat flow out from TI[n].

5.4.3 Problem Splitter Function

At the pinch temperature, the problem will be separated into two subproblems. The function called Program Splitter will separate ROOTNODE to be HOTROOT node and COLDROOT node. All streams in each node will be recalculated their new heat load. If heat load of a stream is zero the status of a stream will be set as "N/A". Otherwise, "ACTIVE" will be set to a stream.

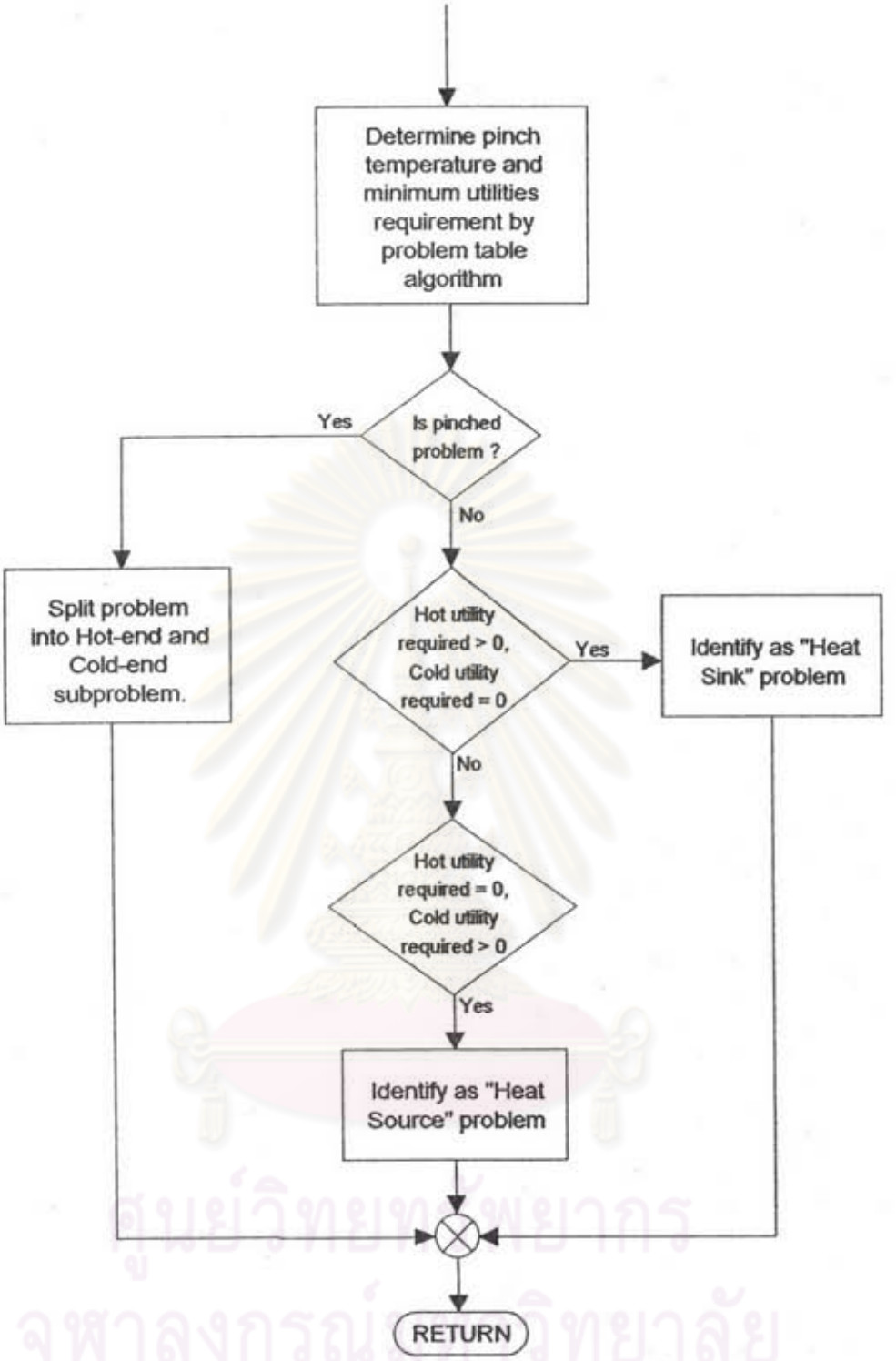

Figure 5.2 Simplified flow chart of the Preanalysis Step.

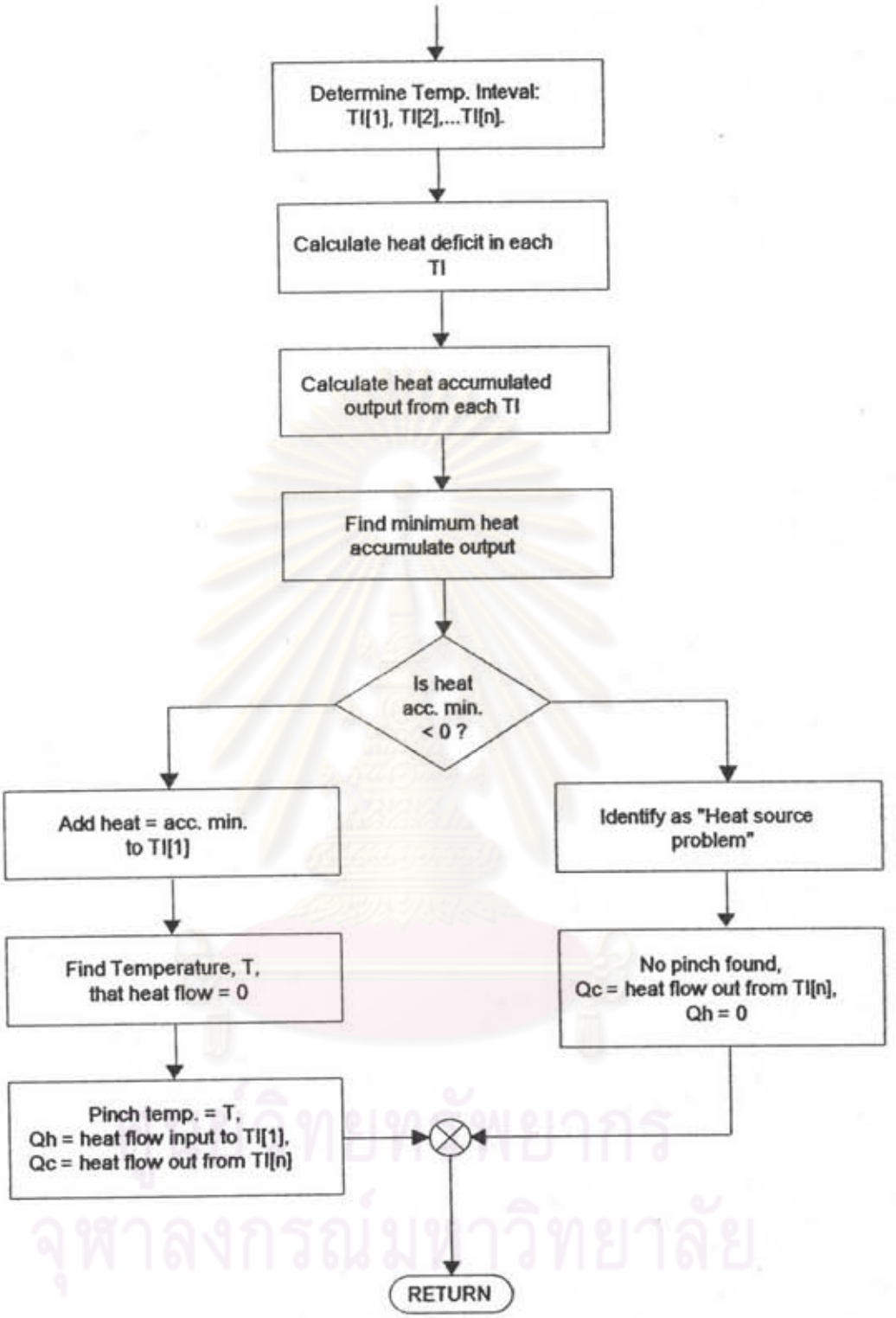

Figure 5.3 Flow chart for determining pinch temperature and minimum utility requirement.

5.4.4 Network Inventer Function

Figure 5.4 shows the simplified flow chart for the network invention step. The function will separately solve each subproblem by using corresponding subnetwork design functions. The subnetwork design functions will use match patterns or match operators to match streams together. At each matching state, the function will create a node to keep the status and the actions in that state. The function will repeat matching until the subsolution is found and it will backtrack to a previous state to try to find other solutions. The networks which are combined by a user will be checked for the existence of heat exchanger loops. The function can find the loop up to second level. If the loop exists, the function will show a message. A user can either choose another subnetwork to avoid that heat loop or break the loops manually.

5.4.5 Designer Function

A heat exchanger network is a search problem as stated in chapter 4. The function flow chart is shown in Fig. 5.5. The designer program will start action from the subproblem root nodes, HOTROOT or COLDROOT, and will set root node to be NODE[0]. The match pattern or match operator at the top of the match pattern stack will be selected to match streams (see Fig. 5.6). If a match is found, the designer will generate a new node, NODE[1], and update necessary information from the parent node, NODE[0], to the child node, NODE[1]. But if the current match operator can not find a match, the next match operator in the stack will be selected to find a pair of the hot and cold streams that satisfy its conditions.

After matching and updating the information, if the status of any stream in a node is still active, the node status will be set to OPEN and the designer will continue searching for the avaiable match. If the current match operator find a match, the designer will create a new node, a child node of the current node, and update all necessary information for the new node. The searching will continue until all of streams are matched and then the latest

node status will be set to GOAL, i.e., a solution is found. To find the others solution, the designer must backtrack to the previous node , NODE[i-1], and use the current match operator recorded of that node. Therefore, a parent node can have more than one child node but a child node can not.

In any design state, if all of match operators are used and no match is found, the node status will be set to CLOSED. The designer will backtrack its parent state to find the available matches in that state.

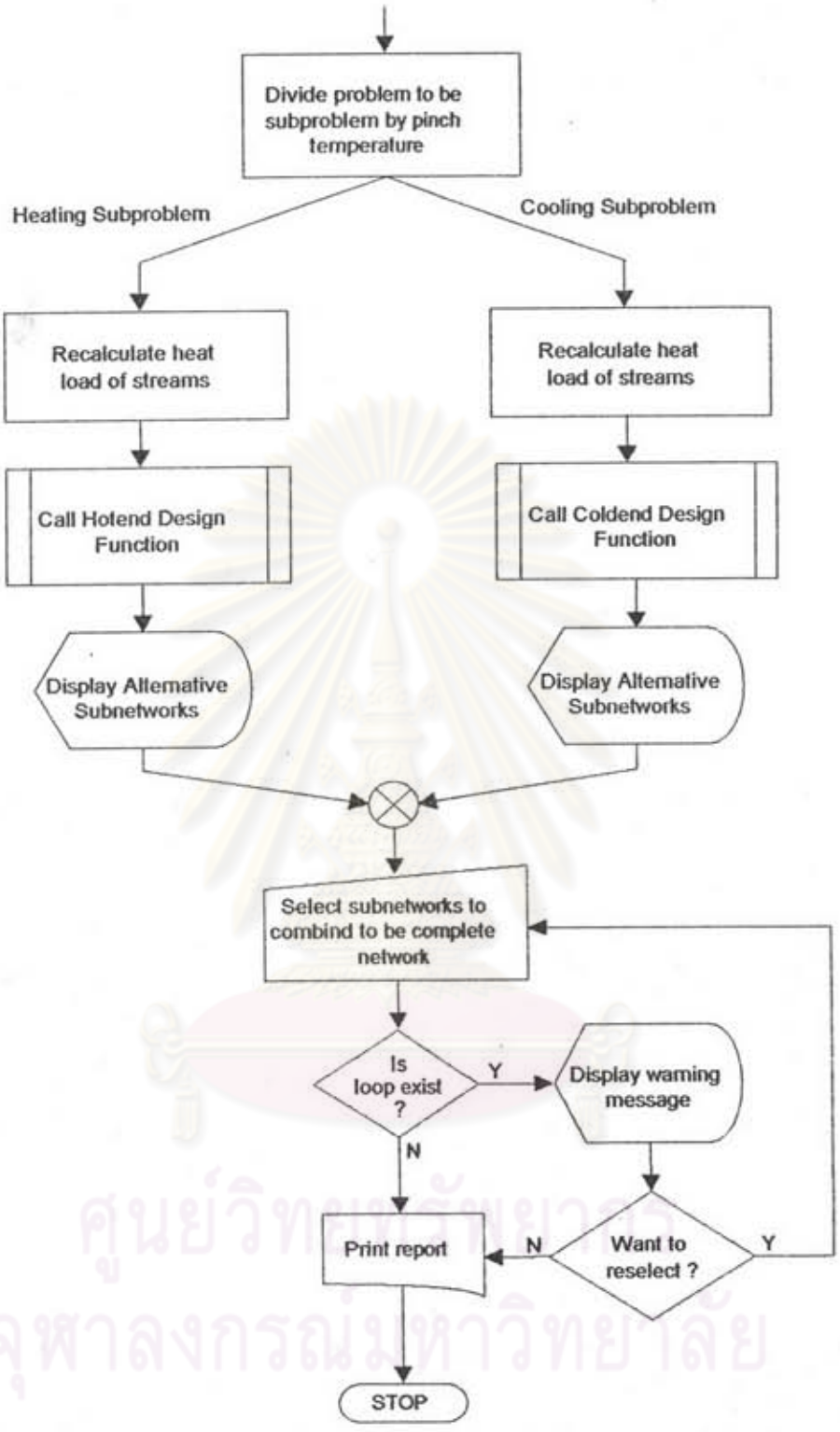

Figure 5.4 A simplified flow chart of the Network Invention Step.

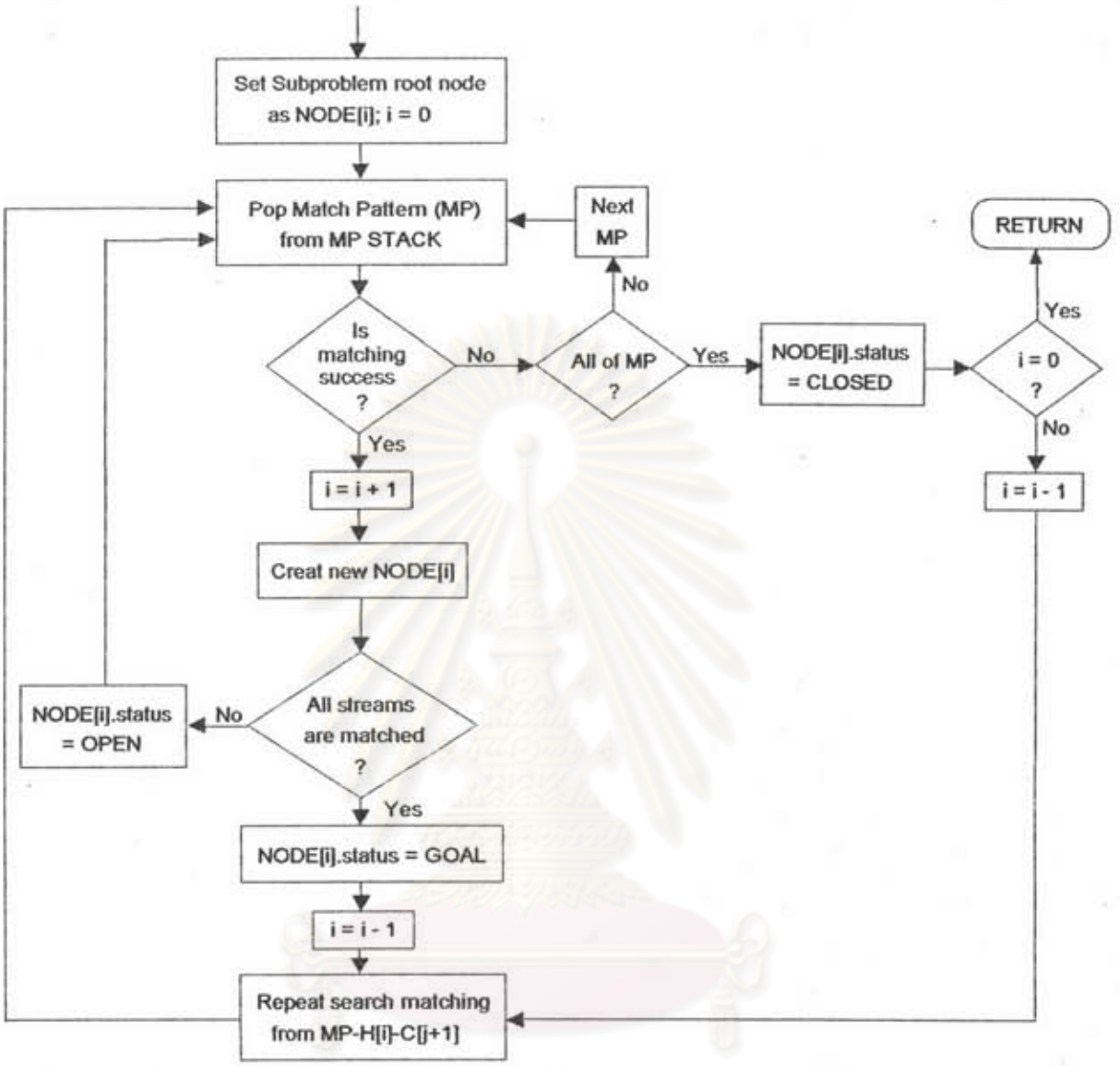

Figure 5.5 A flow chart of the Designer function

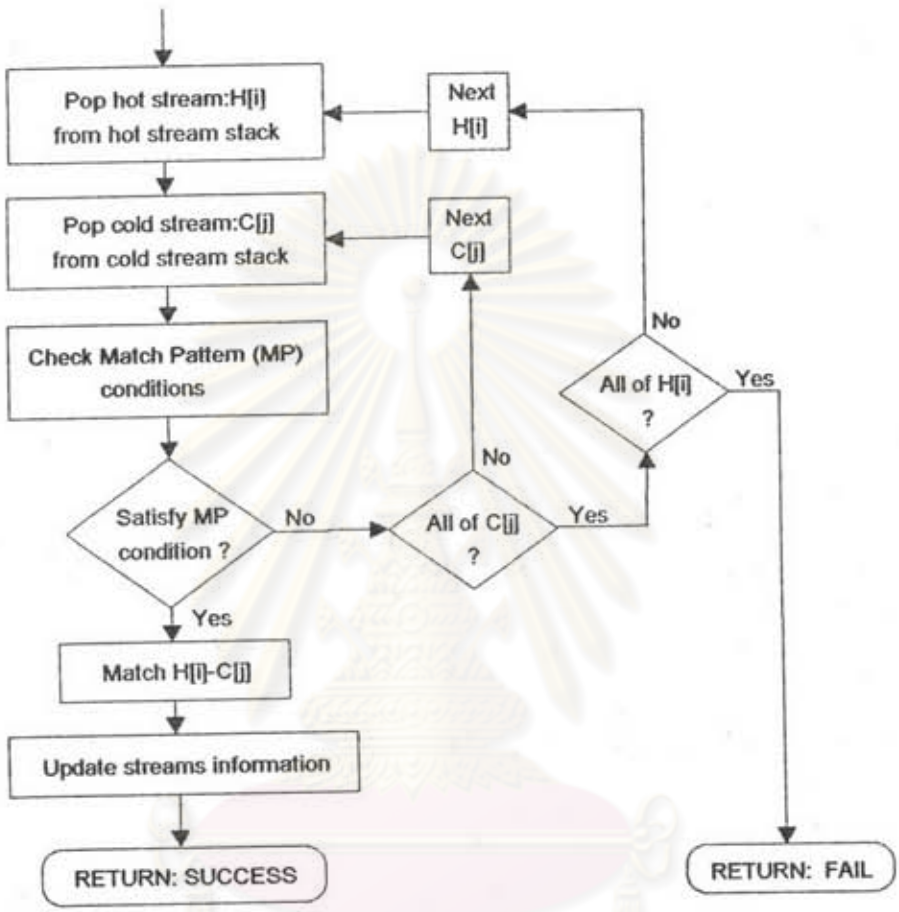

Figure 5.6 A flow chart showing how match operators work (matching process).