**7.2.2 Define problem.**

**-** User request four points, start point, end point, two middle points and their settings includes: walking distance, transfer turn and not optimize option. System get requested points and settings, then search for suitable routes, for user travel from start point to end point though two middle points.

**7.2.3 Attribute Definition.**

* COMBINATIONS: represent list of element which combination from two point adjacent.
* Journey: represent the journey from start point to end point though middle points, includes: total distance, time.

**7.*2*.4 Solution**.

* Multi points search route base on “Two points algorithm”.
* To solve problem, we follow these steps:
  + Step 1: Swap two point of middle points, then we have two results represent list of point will be passed. Suppose four points are A, B, C, D so we have 2 results are ABCD, ACBD.
  + Step 2: With two result above. We exhaust two points of four points of each result, each point and the next point will be combined to an element. And each combined element will be added into list of element we call it is COMBINATIONS. Suppose four inputted points from user are A, B, C, D, so COMBINATIONS is AB, BC, and CD.
  + Step 3: We use “two points algorithm” to solve each element of COMBINATIONS. Each element is include LIST\_RESULTS – the attribute from algorithm.
  + Step 4: Build list of journey, we call it is LIST\_JOURNEYS. Each journey will be built from COMBINATIONS and includes:
    - Total time: total of time from each result of LIST\_RESULT of each element in COMBINATIONS.
    - Total distance: total of distance from each result of LIST\_RESULT each element of COMBINATIONS.
  + Step 5: Sort LIST\_JOURNEYS follow two priorities:
    - Time is high priority. We sort the list base on increasing time.
    - Distance is low priority. We sort the list base on increasing distance.
  + Step 6: With LIST\_JOURNEYS sorted, we get first six element for show to user.