HCI: Dialog Design

Learning Objective

- One important aspect of HCI is the dialog, which is the interaction that takes place between a human user and the computer
- In this lecture, we shall learn about representation, modelling and analysis of dialogs

Dialog

- A dialog refers to the *structure* of the interaction
- Dialog in HCI can be analyzed at three levels
  - Lexical: at this level, details such as the shape of icons, actual keys pressed etc. are dealt with
  - Syntactic: the order of inputs and outputs in an interaction are described at this level
  - Semantic: the effect a dialog has on the internal application/data is the subject matter at this level

Dialog Representation

- We need (formal) techniques to represent dialogs, which serves two purposes
  - It helps to understand the proposed design better
  - Formal representation makes it possible to analyze dialogs to identify usability problems (e.g., we can answer questions such as "does the design *actually* support *undo*"?)

Dialog Representation

- There are several formalism that we can use to represent dialogs
- We shall discuss three of these formalisms in this lecture
  - The state transition networks (STN)
  - The state charts
  - The (classical) Petri-Nets

State Transition Network (STN)

- STNs are the most intuitive among all formalisms
- It assumes that a dialog essentially refers to a progression from one state of the system to the next in the system state space (in fact this assumption holds for all formalisms that we shall discuss)

State Transition Network (STN)

- The syntax of an STN is simple and consists of the following two entities
  - Circles: a circle in a STN refers to a state of the system, which is labeled (by giving a name to the state)
  - Arcs: the circles are connected with arcs, each of which refers to the action/event (represented by arc labels) that results in the system making a transition from the state where the arc originates to the state where it terminates

State Transition Network (STN)

Let’s illustrate the idea with an example. Suppose, we are using a drawing interface that allows us to draw lines and circles, by choosing appropriate menu item. To draw a circle, we need to select a center and circumference. A line can be drawn by selecting points on the line. How can we model this dialog using an STN?

State Transition Network (STN)

- So, what are states and transitions here?
  - We shall have a “start” state
  - From this “start” state, we shall go to a “menu” state, where we are shown the menu options. If we select the circle option, we go to a “circle” state. Otherwise, we select the “line” option and go to the “line” state

State Transition Network (STN)

- So, what are states and transitions here?
  - While at the “circle” state, we select a point as the circle center (through mouse click, say), which takes us to the “center” state
  - In the “center” state, we select the circle periphery (through mouse movement, say) and double click to indicate the end of input (the “finish” state). At this stage, the circle is displayed

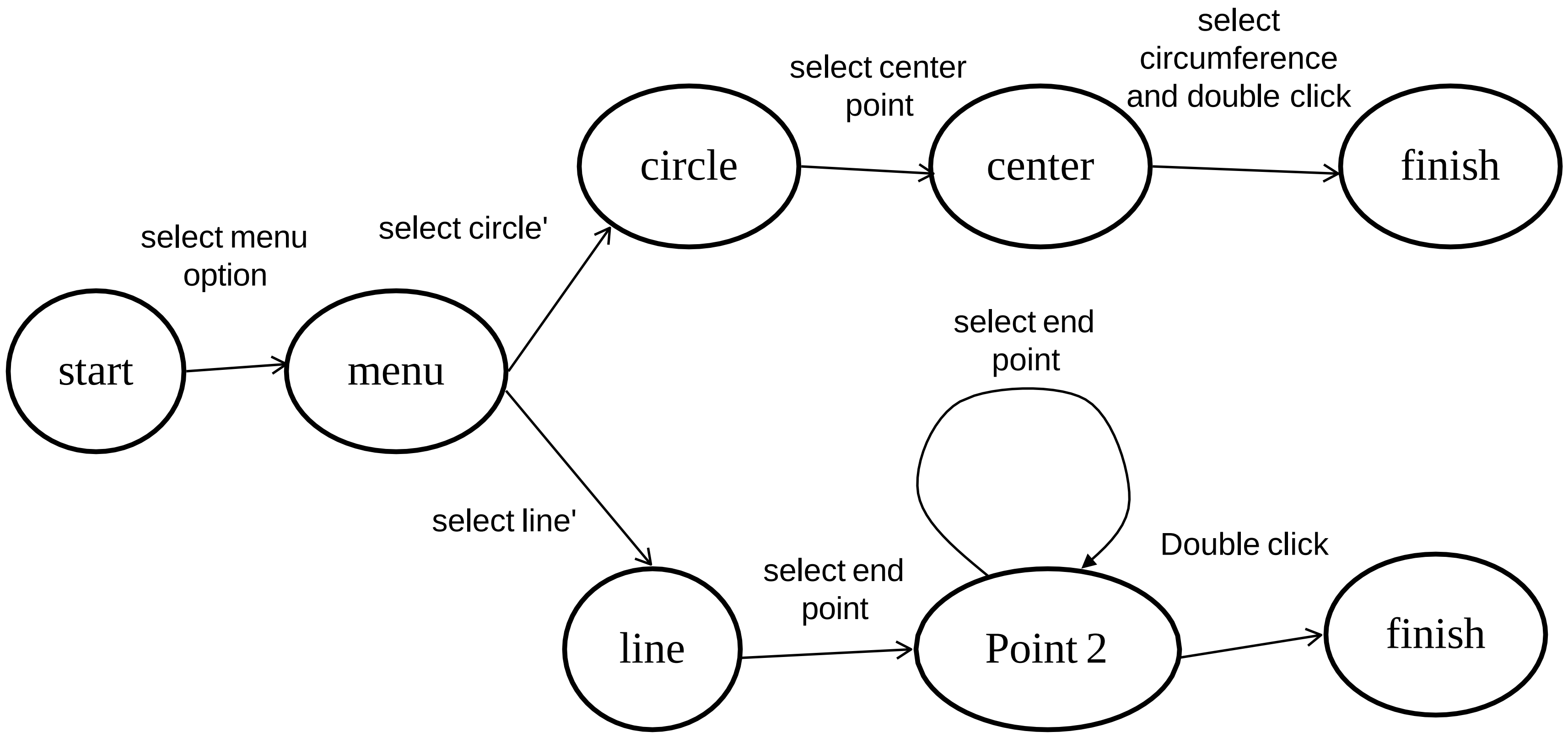
State Transition Network (STN)

- So, what are states and transitions here?
  - While at the “line” state, we select a point as the beginning of the line (through mouse click, say)
  - Then, we select another point to denote the last point on the line and transit to “point 2”. At this stage, a line is displayed between the two points

State Transition Network (STN)

- So, what are states and transitions here?
  - We can select another point, while at “point 2” to draw another line segment between this point and the point last selected. We can actually repeat this as many times as we want, to draw line of arbitrary shape and size
  - When we perform a double click, it indicates the end of input and the dialog comes to the “finish” stage

State Transition Network (STN)



STN – Pros and Cons

- Pros
  - Intuitive
  - Easy to understand
- Cons
  - Good for simple systems
  - Quickly becomes messy as the number of states/arcs grow

How to Model Complex Dialogs

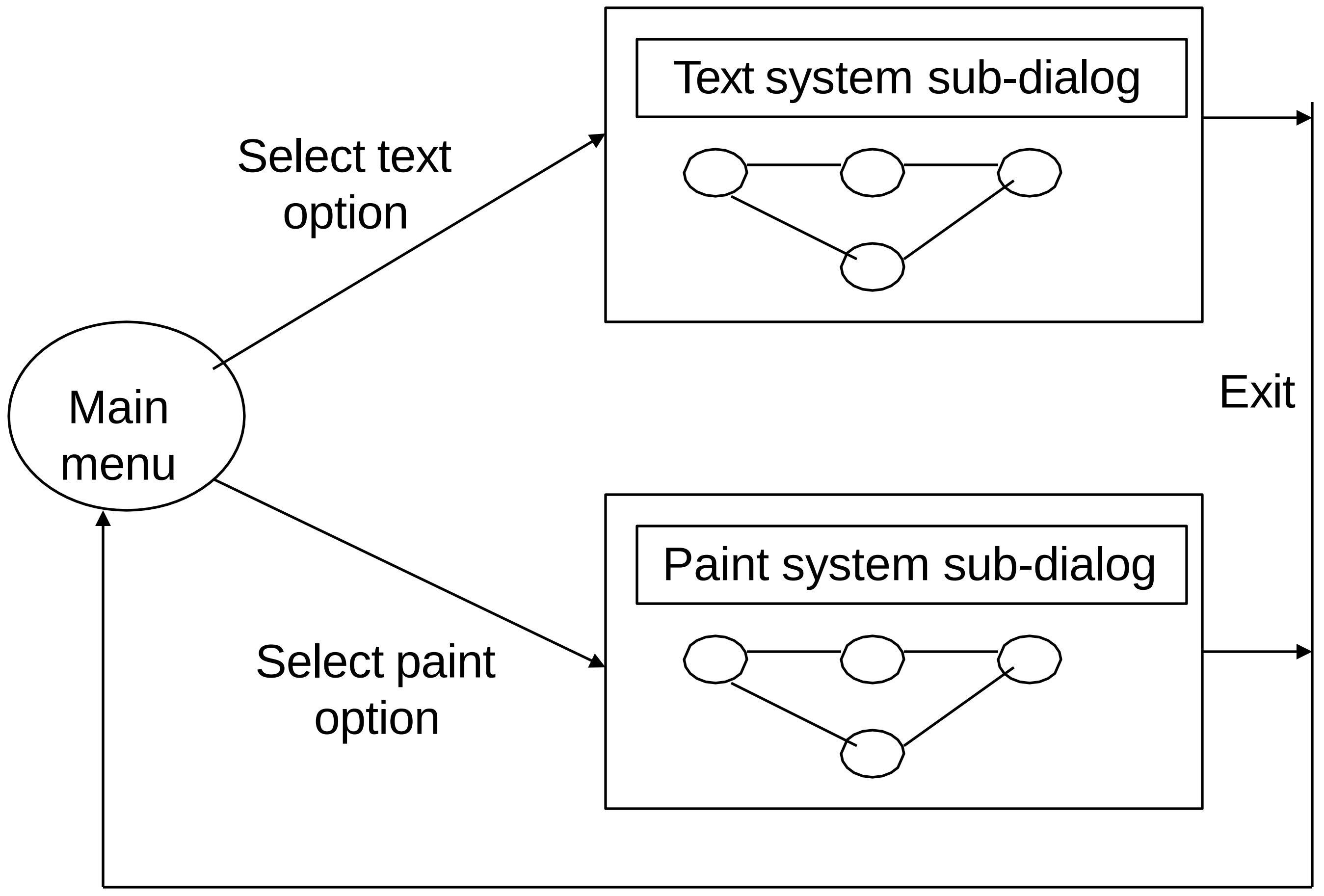
- Hierarchical STNs provide a way to manage complex dialogs
- Here, we divide the dialog into sub-dialogs
- Each sub-dialog is modeled with STNs
- Upper level STNs are designed to connect sub-dialogs

Hierarchical STN - Example

Suppose we want to model the dialog for a menu-based system. There are two menu items, one for a text system and the other for a paint-like system.

Each of these systems has its own dialog. For example, the paint system may have the dialog shown in the previous example. We can model the overall dialog as a hierarchical STN, as shown in the next slide

Hierarchical STN - Example

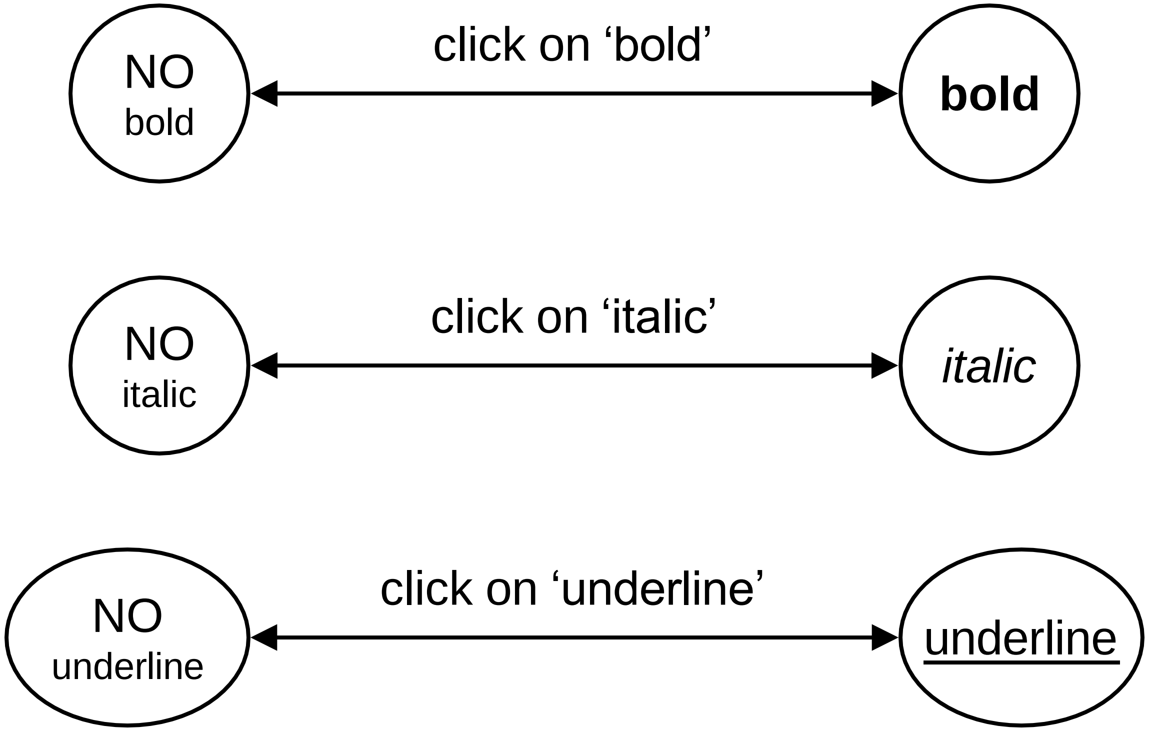


How to Model Complex Dialogs

- However, even hierarchical STNs are inadequate to model many “common” dialogs

  For example, consider a text editor that supports three operations: <u>underline</u>, **bold** and *italic*. Let us try to model this dialog with an STN, assuming first that we can perform (only) one operation on a piece of text

Modelling Complex Dialogs

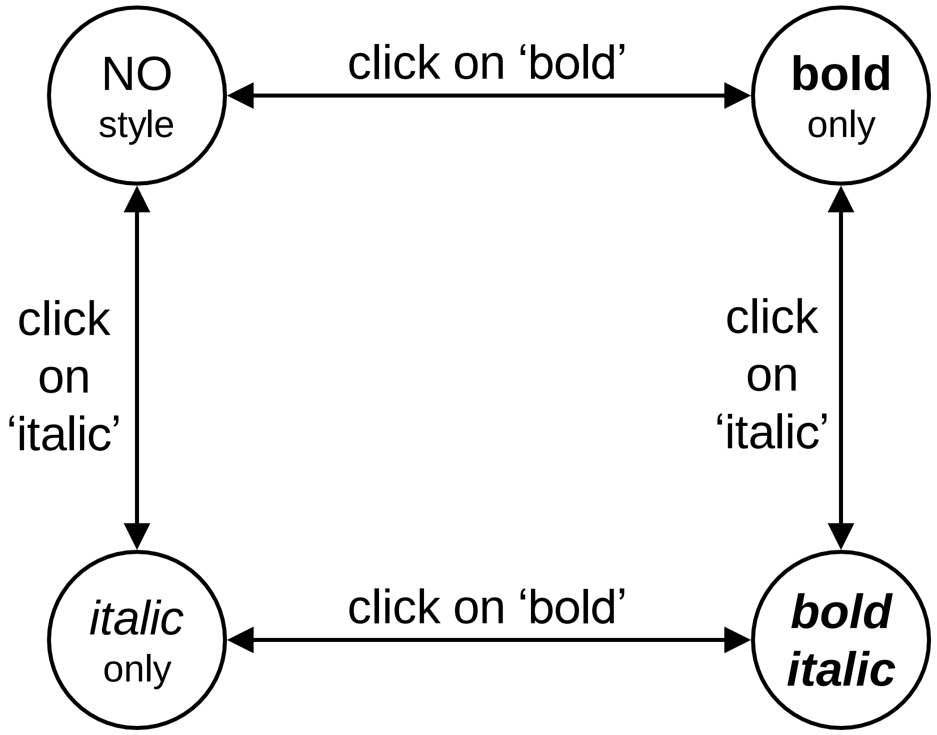


How to Model Complex Dialogs

Now suppose we relax the condition “we can perform (only) one operation on a piece of text”. Now we can perform two operations together on the same piece of text (e.g., **bold** followed by *italic*).
How the STN for this new system looks?

(let us construct the STN for only the dialog involving **bold** and *italic*. STN for other pairs will be similar)

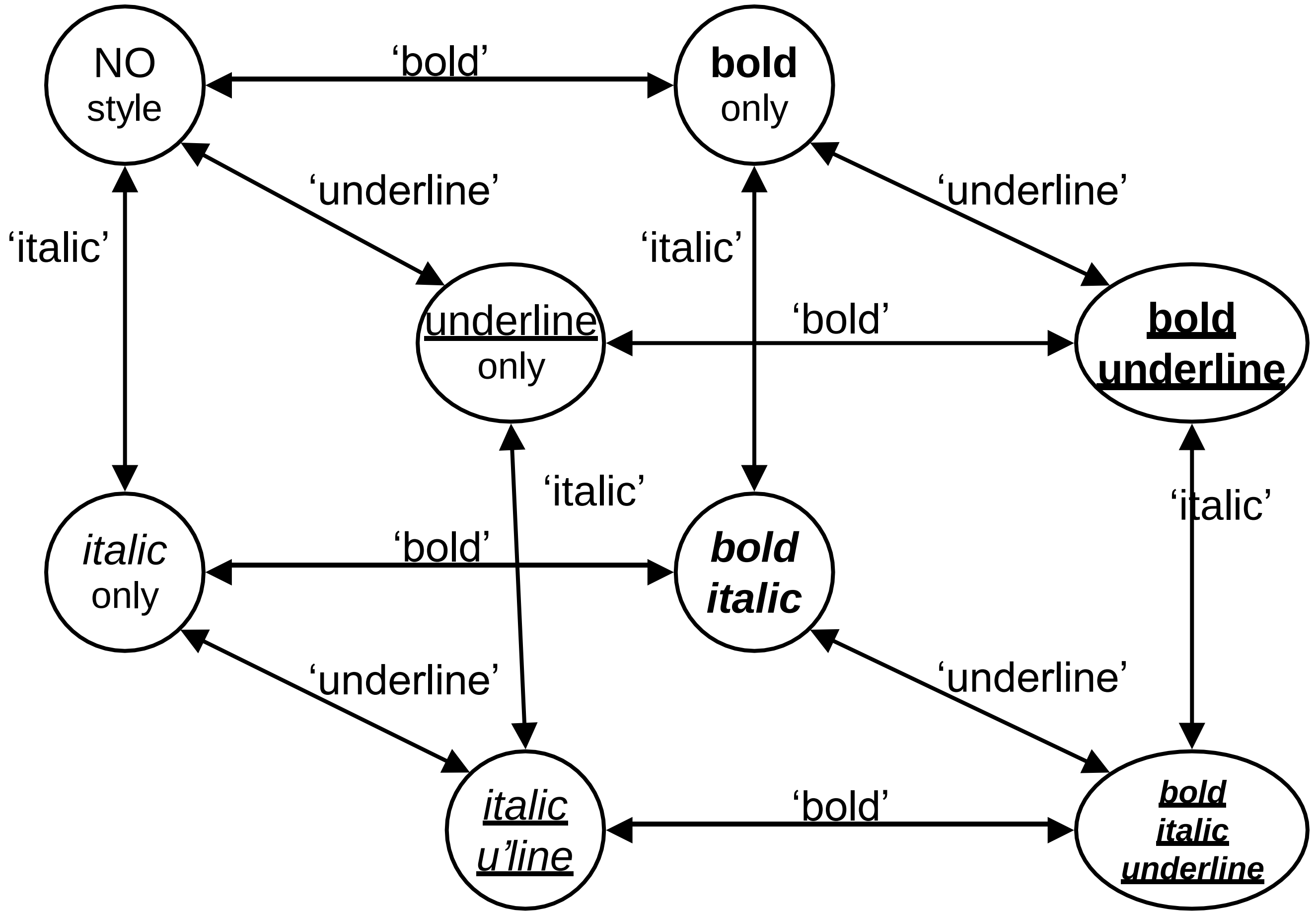
Modelling Complex Dialogs



How to Model Complex Dialogs

Now suppose we relax the condition further. Now we can perform all the three operations together on the same piece of text. This is a fairly common scenario and supported by all text editors. Let us see how the STN for this new system looks.

Modelling Complex Dialogs



How to Model Complex Dialogs

- As we can see, the STN has become very complex with too many states and arcs
- This is because we are trying to model activities that occur on the same object or at the same time. Such behaviors are known as “concurrent behaviors”
- STNs are not very good at modeling concurrent behaviors, which are fairly common in dialogs that we encounter in HCI

Summary

- To better model “concurrent” dialogs, other formalisms are used
- We shall discuss two of those formalisms, namely the State-Charts and the Petri Nets, in the following lectures