Computational Models

- The concept of a computational model
- Basic computational models
- The von Neumann computational model
- Key concepts relating to computational models

TECH
Computer Science

The concept of a computational model

- Model: Foundation or paradigm
- Level of abstraction
- Computational Model
  - **Computer architecture**
  - **Computer language**

Interpretation of concept of a computational model

- Computational Model
  - **(1) Basic items of computation**
  - **(2) Problem description model**
  - **(3) Execution model**

(1) Basic items of computation

- **e.g. data, object, argument and functions, element of sets and the predicates**

(2) Problem description model

- Problem description model
  - **Style**
  - **Method**
- Problem description style
  - **Procedural**
  - **Declarative**
- Procedure style
  - (algorithm for solving the problem is stated)
- Declarative style
  - (all the facts and relationships relevant to the given problem is stated)

Problem description style (e.g.)

Calculate n factorial, n!

- Procedural style
- int nfac (int n) {

```
int fac = 1;
if (n > 0)
  for ( int i = 2; i <= n; i++ )
    fac = fac * i;
return fac; }
```

- Declarative style

```
fac (0) = 1;
fac ( n>0 ) = n * fac ( n-1 );
```

Declarative style

- Using functions
  - in a model called applicative, (Pure Lisp)
- Using predicates
  - in a model called predicate logic-based, (Prolog)

Problem description method

- Procedural method
  - how a solution of the given problem has to be described
  - e.g. sequence of instructions
- Declarative method
  - how the problem itself has to be described
  - e.g. set of functions

(3) Execution Model

- Interpretation of how to perform the computation
  - related to the problem description method
- Execution semantics
  - rule that prescribes how a single execution step is to be performed
- Control of the execution sequence
  - ordering of execution sequence

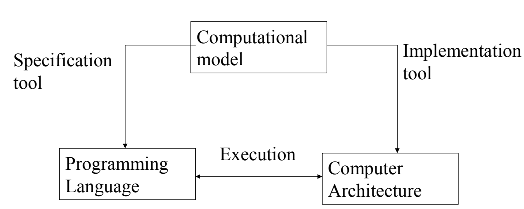
Execution semantic

- State transition semantics
  - Turing model
  - von Neumann model
  - object-based model
- Dataflow semantics
  - dataflow model
- Reduction semantics
  - applicative model (Pure Lisp)
- SLD-resolution
  - Predicate logic-based model (Prolog)

Control of the execution sequence

- Control driven
  - **assumed that there exists a program consisting of sequence of instructions**
    - execution sequence is then implicitly given by the order of the instruction
    - explicit control instructions to change the order
- Data driven
  - **an operation is activated as soon as all the needed input data is available (eager evaluation)**
- Demand driven
  - **an operation is activated only when execution is needed to achieve the final result**

Concepts of computational model, programming language, and architecture

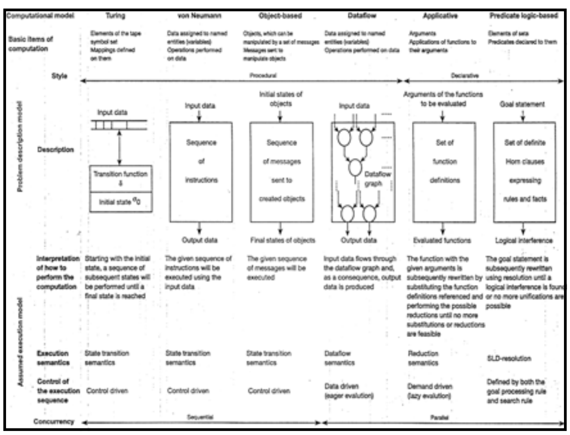


Typical Evolution

- Computation model
- Corresponding programming language
- Corresponding architecture

Basic computational models

- Turing
- von Neumann
- object based
- dataflow
- applicative
- predicate logic based



Computational model	Turing	von Neumann	Object-based	Dataflow	Applicative	Predicate logic-based
Basic items of computation	Elements of the tape symbol set Mappings defined on them	Data assigned to named entities (variables) Operations performed on data	Objects, which can be manipulated by a set of messages Messages sent to manipulate objects	Data assigned to named entities (variables) Operations performed on data	Arguments Applications of functions to their arguments	Elements of sets Predicates declared to them
Style	Procedural				Declarative	
Description (Problem description model)	Input data; Transition function δ; Initial state q_0	Input data → Sequence of instructions → Output data	Initial states of objects → Sequence of messages sent to created objects → Final states of objects	Input data → Dataflow graph → Output data	Arguments of the functions to be evaluated → Set of function definitions → Evaluated functions	Goal statement → Set of definite Horn clauses expressing rules and facts → Logical interference
Interpretation of how to perform the computation	Starting with the initial state, a sequence of subsequent states will be performed until a final state is reached	The given sequence of instructions will be executed using the input data	The given sequence of messages will be executed	Input data flows through the dataflow graph and, as a consequence, output data is produced	The function with the given arguments is subsequently rewritten by substituting the function definitions referenced and performing the possible reductions until no more substitutions or reductions are feasible	The goal statement is subsequently rewritten using resolution until a logical interference is found or no more unifications are possible
Execution semantics (Assumed execution model)	State transition semantics	State transition semantics	State transition semantics	Dataflow semantics	Reduction semantics	SLD-resolution
Control of the execution sequence	Control driven	Control driven	Control driven	Data driven (eager evalution)	Demand driven (lazy evalution)	Defined by both the goal processing rule and search rule
Concurrency	Sequential			Parallel		

The von Neumann computational model

- Basic items of computation are **data**
  - **variables (named data entities)**
  - **memory or register locations whose addresses correspond to the names of the variables**
  - **data container**
  - **multiple assignments of data to variables are allowed**
- Problem description model is **procedural** (sequence of instructions)
- Execution model is state transition semantics
  - **Finite State Machine**

von Neumann model vs. finite state machine

- **As far as execution is concerned the von Neumann model behaves like a finite state machine (FSM)**
- FSM = $\{ I, G, \delta, G_0, G_f \}$
- I: the input alphabet, given as the set of the instructions
- G: the set of the state (global), data state space D, control state space C, flags state space F, $G = D \times C \times F$
- δ: the transition function: $\delta: I \times G \rightarrow G$
- G_0 : the initial state
- G_f : the final state

Key characteristics of the von Neumann model

- Consequences of multiple assignments of data
  - **history sensitive**
  - **side effects**
- Consequences of control-driven execution
  - **computation is basically a sequential one**
- ++ easily be implemented
- Related language
  - **allow declaration of variables with multiple assignments**
  - **provide a proper set of control statements to implement the control-driven mode of execution**

Extensions of the von Neumann computational model

- new abstraction of parallel execution
- communication mechanism allows the transfer of data between executable units
  - **unprotected shared (global) variables**
  - **shared variables protected by modules or monitors**
  - **message passing, and**
  - **rendezvous**
- synchronization mechanism
  - **semaphores**
  - **signals**
  - **events**
  - **queues**
  - **barrier synchronization**

Key concepts relating to computational models

- Granularity
  - **complexity of the items of computation**
  - **size**
  - **fine-grained**
  - **middle-grained**
  - **coarse-grained**
- Typing
  - **data based type ~ Tagged**
  - **object based type (object classes)**