Seminar W4: Foundations for Programming

W4.1. Programming as prescription & interpretation of state-change.

What is programming? What is parallel programming?

W4.1.1. Classical one-person programming

[Here "one-person programming" interpreted as "programmer = user"]

User role in user-computer interaction

- User identifies a computational object
  - User can observe changes in state of an object
  - User can initiate changes of state
  - User has expectations that are reliably confirmed
- User/programmer devises pattern of initiation and observation of state
- User supplies input according to pre-conceived pattern
- User interprets the observations

Classical framework

User identifies a computational object

```
>   HERE IS A SUN WorkStation
```

User can observe changes in state of an object

```
>   SUN WorkStation conveys info through textual/graphical output
>   primarily - though does make noises from time to time
```

User can initiate changes of state

```
>   SUN WorkStation can be programmed to accept input
>   in form of text or mouse actions
```

User has expectations that are reliably confirmed

```
>   SUN WorkStation does what you expect of it reliably
```

User/programmer devises pattern of initiation and observation of state

```
>   User writes a program to accept input and give output
>   according to some preconceived pattern of interaction
```

User supplies input according to pre-conceived pattern

```
>   User gives input to the program when it is requested
```

User interprets the observations

> User refers to output, possibly to influence next input
> Perhaps - as programmer - to influence next version of the program
> "use", "debug" or "redesign" program

Points

Sequential: (initiate change of state, observe change of state)*

Variants on the degree and nature of interpretation involved:
e.g. pure function computation : interpret IO only
    simulation : might observe many programmed state changes closely

Terminology and Key Principles

User and computer AGENTS : capable of causing state-changes
    cf dictionary definition of agent
Changes of state that an agent can initiate PRIVILEGES
    Privileges are context specific: e.g. user-input at given time

DESIGN vs SIMULATION distinction wrt interpretative activity
Have conflated two roles: user / programmer to emphasise
    programmer designs, user runs simulation / executes program

W4.1.2. Computational Objects and Computational models

Evidently don't need a SUN Workstation to do computation
    e.g. Turing Machine, Babbage, multiply by 9, GCD folk-dance

Computational object?
    reliable pattern of autonomous state-change / observation
    conventions for observation (cf folk-dance:
        WHEN everyone's paired off
        can't create new pairs, nobody drops dead
        resource implications: e.g. realistic to observe SO MANY people?
)

Reliability is a matter for experiment

In the experiment, key issues are
    1) synchronisation of observation
        what do we choose to regard as synchronised observation?
    2) conventions re experimental observation
        what do we choose to observe? - defines what is meant by state
        what assumptions do we make about potential interference?

For the SUN WorkStation, have been through the process of experiment
e.g. registers are read after they have stabilised electrically,
    the user is insulated from intermediate states in execution

For a non-standard device (such as folk-dance computer), the conventions
are more open to question and change
e.g. how fast will people pair up?
        how would other activities interfere? (people eat / sleep etc)
        what perception / capabilities do people need to perform the dance?
        [e.g. must communicate to everybody when all pairs are identified:
        can people see & hear, can they decide for themselves when to act?]

Developing a program involves administration of action:
    conventional machines: actions reliable, circumscribed
                                            administration "easy"
    unorthodox machines: actions complex, subject to interference
                                            administration complex

Parallel programming is special case of computation on unorthodox m/c.

Terminology and Key Principles

CONTEXT for observation : "correlate observations in the same context"
Correlation of observation is intimately connected with synchronisation

W4.2 Experiment and Observation

Consider a simple experiment, such as studying Hookes' Law

Context for observation is defined by a (mass, extension) pair
These might not be measured at the same instant of time, BUT
    can't measure one extension and another mass
    must assume that "nothing interferes" between two measurements

No other means to decide whether there has been interference
than to consult observations and expectations

Interference has to be understood with reference to prior knowledge
of observations : "ordinarily I would expect this to be observed"

[20th century variant of older experiment: assume the masses are magnetised generate a magnetic field to attract the masses and vary the field as the experimentor changes the masses. An experimentor who doesn't know about magnetism can't verify Hookes' Law.]

Categories of knowledge and associated concepts

Imagine experiment whereby change o1 and observe o2

I know how to relate observation o1 to o2 : good experiment
I know how o1 and o2 would be related were no other factor involved :
    something else is interfering : another agent is present

So far as I can see, o1 and o2 are quite independent observations :
    many agents are involved : not useful experiment

Extent to which can recognise dependencies between observations determines extent to which changing observation o1 bears interpretation

[Many state-changing acts we can perform appear to have no significance]

The "Good Experiment" Paradox

1) A "Good Experiment" is one where we always get the result we expect

    [These are the experiments every physics student does: BUT
    of course *in principle* such an experiment might fail!]

2) A "Good Experiment" is one where we don't know what to expect

    [This is what is ordinarily meant by an experiment: BUT
    of course there has to be an idea of what to observe and expect]

In 1) reliable activity is the key: emphasis on the "computational object"
In 2) interesting interpretation is the key: emphasis on the "user"

Activity 1) reassures us about the computational object
Activity 2) guides us towards a more satisfactory interpretation

Both experiment 1) and 2) may involve exactly the same observations
    Appeal to a pattern of observations in 1) as *confirmation of faith*
    Appeal to a pattern of observations in 2) as *basis for belief*
Two activities are on different sides of an act of faith.

Pre act-of-faith is Requirements modelling phase activity:
    concerned with clarifying what we should believe to work
Post act-of-faith is Implementation activity:
    concerned with what we reliably expect to happen

This is Modelling vs Programming
    Programming is one kind of interpretation for modelling activity
    Computation is one kind of interpretation for experimental activity

Observation is a primitive abstraction neutral wrt to the act-of-faith
    Observation needs no presumption about circumscribed behaviour
    cf object modelling or theory building
Temptation to associate observations with objects and theories: proper separation of concerns distinguishes two aspects of observation
    objective measurement vs meaningful measurement
e.g. observe bird fly from a building to another, but *know* no reason why, and - in some modes of observation - don't even *consider* why

Characteristic of good experiment is correlation between observations
[whichever interpretation we choose?]

W4.3. Variables and Correlation between Observations

Correlation between observations in the same context:
given observation o1, I can predict observation o2

Key issues:

FD There is an independent method for deriving o2 from o1.
i.e. there is an independent experiment that realises o2 from o1.
In particular, is a ("generalised") computation to realise o2 from o1.

PV Observation and context for observation together make framework
for *identity* and *change*. To perform experiment, must presume
variables of this nature: need to measure the same quantity in
a different context.

FD and PV are fundamental abstractions:
both with subtly different modern / historical overtones
FD is something like "a functional dependency"
PV is something like "a procedural variable"

Qualifications:

Modern use of functional dependency concept is very liberal.
e.g. a functional program strains the concept of dependence
between observations made in the same context. In contrast,
FD is the historically primary use of the concept of function.

<Russian quotation>

Procedural variables are associated with modelling activity
very loosely associated with external observation, as e.g. in
the use of identifiers in a conventional procedural program.
In contrast, PV is directly concerned with external reference.

PV also differs emphatically from the modern mathematical variable:
though it is the historically primary use of the concept of variable.

<quotation from Bird and Wadler + ... >

Other issues:

FD is to be distinguished from "a logical relation between o1 and o2"
a) there's an asymmetry between o1 and o2

b) o1 and o2 are not to be regarded as logical variables.
[cf concern about whether observations presume "theory".]

Correlation between observations is modelled by representing FDs

We do this
1) by introducing variables to represent observations
2) by specifying the primitive constructions that can feature in the independent experiments cited in FD above
3) by recording the formula that can be used to derive o2 from o1

Formalise this in *definitive scripts*
1) => special kind of variable "definitive" (cf PV above)
2) => basic repertoire of computational types and operators
these make up an "underlying algebra"
3) => introduce definition o2=f(o1) to specify recipe to get o2 from o1

Note the orthodox nature of the computational model in 2)
cf ADTs, functional programs, encapsulated objects.

"From modelling to programming" is process of relating
- uncircumscribed observation (pre-act-of-faith) to
- circumscribed class of reliable experiments (post-act-of-faith).