Goals

• help tool-builders
  – to deal with true concurrency
  – and to interwork with other tools

• help experimenters
  – to understand the tools
  – and to use them more effectively

• delight all of you
  – by the simplicity, power and elegance of algebra
1. Laws of programming

Axioms describe algebraic properties of each operator
- association, commutation, idempotence
- distribution laws for pairs of operators
- units and zeroes (constants, with no operands)
Notations

• then ; sequential composition
• with || concurrent composition
• skip I does nothing

Relation

• definition => if the lhs is defined, it is equal to the rhs
Axioms

• **assoc** \[ p;(q;r) = (p;q);r \] (also ||)

• **comm** \[ p\|q = q\|p \]

• **unit** \[ p\|I = p = I\|p \] (also ;)
Reversibility

- **assoc** \( p;(q;r) = (p;q);r \) (also \( || \))
- **comm** \( p||q = q||p \)
- **unit** \( p||I = p = I||p \) (also \( ; \))

- reversal of the order of operands of either operator translates one axiom into another.
- and translates the associative axiom into itself.
Reversibility

• Metatheorem: (duality)
When a theorem is translated by reversing the operands of all \( \land \)s (or of all \( \lor \)s), the result is also a theorem.

• Many laws of physics are reversible
Linearity

- **assoc** \[ p;(q;r) = (p;q);r \] (also ||)
- **comm** \[ p||q = q||p \]
- **unit** \[ p||I = p = I||p \] (also ;)

- each axiom contains all its free variables twice: once on each side of the = .
- such equations (or inequations) are called linear
Axiom

• => is a partial order
  – reflexive        p => p
  – transitive      if p => q & q => r then p => r

• Definition: an operator • is monotonic if
  – p => q        implies        p•r => q•r
    & r•p => r•q
Monotonicity

• Axiom: Both our operators are monotonic
• Metatheorem (theorems for free):
Let $F(p)$ be a formula containing $p$.
Let $F(q)$ result from replacing $p$ by $q$ in $F(p)$
Let $p \Rightarrow q$ be a theorem

Then $F(p) \Rightarrow F(q)$ is also a theorem
Exchange Axiom

• \((p \parallel q) ; (p' \parallel q') \implies (p;p') \parallel (q;q')\)
  – named after the interchange law in categories

• Theorem (frame): \((p \parallel q) ; q' \implies p \parallel (q;q')\)
  – Proof: substitute \(I\) for \(p'\) in exchange axiom

• Theorem: \(p;q \implies p*q\)

• The axiom is reversible and linear
The Hoare triple

• Definition: \( \{p\} q \{r\} = p;q \Rightarrow r \)

• Theorem:

\[
\{p\} q \{s\} \quad \{s\} q' \{r\}
\]

\[
\{p\} q;q' \{r\}
\]

Proof: \( ; \) is monotonic and associative
Proof in full

• Assume $p;q \Rightarrow s$ and $s;q' \Rightarrow r$

• by mon $(p;q);q' \Rightarrow r$

• by assoc $p;(q;q') \Rightarrow r$

• by def $\{p\} q;q' \{r\}$

• proof of the other laws of Hoare logic are equally simple
Modularity rule for $\|\|

\{p\} q \{r\} \quad \{p'\} q' \{r'\}
\{p || p'\} q || q' \{r || r'\}

– permits modular proof of concurrent programs.
– equivalent to exchange law
Modularity implies exchange

• Assume: \( p;q = r \) and \( p';q' = r' \)

• => reflexive: \( p;q \Rightarrow r \) and \( p';q' \Rightarrow r' \)

• modularity: \( (p||p') ; (q||q') \Rightarrow (r||r') \)

• by assumption:
  \( (p||p') ; (q||q') \Rightarrow (p;q) || (p';q') \)

and vice-versa
2. The axioms apply to programs

• Construct a realistic model of program behaviour
  – e.g., distinguishing concurrency from interleaving
• Define the operators in terms of the model
• Prove that the definitions satisfy all the axioms
Program Behaviour

• Let $\top$ be the set of all possible behaviours
  – of all possible programs
  – in all possible circumstances of use

• A program is modelled as a subset of $\top$.
  – containing just the behaviours which result from execution of the program.

• $P, Q, R, P', \ldots$ will stand for specifications
Definitions

- $\bot = \{ \} , \text{ the empty set},$
  - with no behaviours
  - an implementation can reject such a program
  - preferably before it starts

- $P \text{ \setminus Q} = P \cup Q$ \quad set union
- $P \text{ \setcap Q} = P \cap Q$ \quad set intersection
Theorem

• These definitions satisfy all the axioms of a Boolean Algebra.
• Proof: use Venn diagrams
Events

• Let $E$ be the set of all events
  – that can possibly occur in and around a computer
  – during the execution of any program.

• Then a behaviour is a subset of $E$

• $p, q, r, p'$ ... will stand for behaviours
  – usually $p \in P$, $q \in Q$, ...
Definitions

• $p \sim q = p$ and $q$ are defined and $p \cap q = \{\}$

• $p \oplus q = p \cup q$, provided $p \sim q$
  – disjoint union of sets of events
  – undefined if $p$ and $q$ share a member

• in reality: If $P$ and $Q$ are disjoint parts of the same program, then no event is part of an execution of both $P$ and of $Q$. 
Example

• Suppose events are recorded as just numbers.
  – perhaps the times at which events occur
  – by convention we write them in increasing order
• \{8.45, 10.0, 11.15\} \oplus \{9.00, 10.30\} = \{8.45, 9.00, 10.0, 10.30, 11.15\}
• \{8.45, 10.0, 11.15\} \oplus \{10.00\} is undefined
• concurrency here is modelled by interleaving
  – which obeys the same laws!
Lemmas

• \( p \sim q \iff q \sim p \)
• \( p \oplus q = q \oplus p \)
• \( p \sim (q \oplus r) \iff p \sim q \& p \sim r \& q \sim r \)
• \( p \oplus (q \oplus r) = (p \oplus q) \oplus r \)

• Theorem: \( \oplus \) satisfies basic axioms for \( || \)
Sequential composition

• Let \( p \) and \( q \) be (defined) behaviours

• Let \( p \rightarrow q \) means that (in reality) \( p \) can finish before \( q \) begins
  – e.g. nothing in \( p \) depends on anything in \( q \)

• Define \( p; q \) as \( p \oplus q \) provided \( p \rightarrow q \)
Axioms

• \( p \implies q; r \iff p \implies q \land p \implies r \land q \implies r \)
• \( p \implies q; r \iff p \implies q \land p \implies r \land q \implies r \)
• \( p \implies q \implies p \sim q \)
  – if sequential execution is possible, so is concurrent

Theorem

• \( p; q \implies p || q \)
• \( p ; (q ; r) = (p ; q) ; r \)
\[(p \oplus q) ; (p' \oplus q') \Rightarrow (p;p') \oplus (q;q')\]

- When they are defined, both sides are
  \[p \oplus q \oplus p' \oplus q'\]
- The condition for refinement of the lhs are
  - \(p \oplus q) \Rightarrow (p' \oplus q') =\)
    \[p \bowtie q \land p' \bowtie q' \land p \Downarrow p' \land q \Downarrow q' \land p \Downarrow p' \land q \Downarrow p'\]
- and for the rhs:
  - \((p \oplus p') \bowtie (q \oplus q') =\)
    \[p \Downarrow p' \land q \Downarrow q' \land p \bowtie p' \land q \bowtie q' \land p \bowtie q \land p' \bowtie q\]
- the theorem follows from
  - \(p \Downarrow p' \) implies \(p \bowtie p'\)
\[(p \parallel q); (p' \parallel q') \Rightarrow (p;p') \parallel (q;q')\]
Summary

• All the axioms of our algebra are satisfied by the definitions of $\Rightarrow$ and $\oplus$

provided that $\Rightarrow$ satisfies the axioms

$- p \Rightarrow q; r \iff p; q \Rightarrow r$

iff $p \Rightarrow q \land p \Rightarrow r \land q \Rightarrow r$

$- p \Rightarrow q \implies p \not\Rightarrow q$
Lifting to sets

- \( P||Q = \{ r \mid r \Rightarrow p||q \land p \in P \land q \in Q \} \)
- \( P;Q = \{ r \mid r \Rightarrow p;q \land p \in P \land q \in Q \} \)
- \( I = \{ r \mid r \Rightarrow I \} \)
- \( P \Rightarrow Q \iff P \subseteq Q \)

- Note that ; and || are total functions
  – because undefinition leads to emptiness
- all sets are downward closed wrto =>
Big Theorem

• The operators on sets satisfy the same axioms as the operators on elements.
• Proof: all the axioms are linear.
Additional operators

• Definition: $P \lor Q = P \cup Q$
• Definition: $P \land Q = P \cap Q$
• Theorem: $\; \; \land \; \; \land \; \; | \; \; |$ distribute through $\lor$
  – all operators defined by lifting do so.
Summary

• The lifted algebra satisfies all the axioms of a Concurrent Kleene Algebra.