CIS 505 Software Systems
Lecture Note on CSP

Instructor: Insup Lee
Department of Computer and Information Science
University of Pennsylvania

(The slides are originally prepared by U. Sammapun,
based on the CSP book by C.A.R. Hoare)

Communicating Sequential Processes (CSP)

- Capture real world behaviors
  - Chocolate vending machine
- Event (instantaneous actions)
  - coin
  - choc
- Process (behavior patterns of objects)
  - VMC – chocolate vending machine
- Alphabet (set of possible events, denoted as αP)
  - αVMC = {coin, choc}
- STOPα (behavior or process of a broken object with alphabet A)
  - STOPαVMC

Prefix

- Prefix: (x → P)
  - Process with same alphabet as P
  - α(x → P) = αP where x ∈ αP
- Broken chocolate vending machine
  - coin → STOPαVMC
  - (coin → (choc → (coin → (choc → STOPαVMC))))
- Counter board
  - αCRT = {up, right}
  - CRT = (right → up → right → right → STOPαCRT)
- Incorrect
  - P → Q
  - x → y (Correct syntax: x → (y → STOP))

Recursion

- Describe repetitive behaviors
- Clock
  - αCLOCK = {tick}
  - CLOCK = tick → CLOCK
  - CLOCK = tick → (tick → CLOCK)
  - CLOCK = tick → (tick → (tick → CLOCK))
  - CLOCK = tick → (tick → (tick → (tick → CLOCK)))
  - ……
  - = tick → tick → tick → tick → tick → tick → …… [unfolding CLOCK many times]
- Useless equation?
  - X = X
Recursion

• Guarded
  – A process description that begins with a prefix
  – $CLOCK = (\text{tick} \rightarrow CLOCK)$

• If $F(X)$ is a guarded expression containing process $X$ and alphabet $A$
  – Then $X = F(X)$ has a unique solution with alphabet $A$
  – Solution: $\mu X: A \cdot F(X)$

• $X$ is a local name and can be changed
  – $\mu X: A \cdot F(X) = \mu Y: A \cdot F(Y)$
  – Because a solution for $X = F(X)$ is also a solution for $Y = F(Y)$

Recursion Examples

• Clock
  – $CLOCK = \mu X: (\text{tick}) \cdot (\text{tick} \rightarrow X)$

• Working chocolate vending machine!
  – $VMC = (\text{coin} \rightarrow (\text{choc} \rightarrow VMC))$
  – Or formally, $VMC = \mu X: (\text{coin} \cdot \text{choc}) \cdot (\text{coin} \rightarrow (\text{choc} \rightarrow X))$

• Machine gives change for 5p repeatedly
  – $\alpha CH5A = \{\text{in5p}, \text{out2p}, \text{out1p}\}$
  – $CH5A = (\text{in5p} \rightarrow \text{out2p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow CH5A)$

• Different change-giving machine with same alphabet
  – $CH5B = (\text{in5p} \rightarrow \text{out1p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow CH5B)$

Choice

• Interaction with environment
  – Machine with 1p coin slot and 2p coin slot
  – It’s a customer’s choice

• Syntax
  – $(x \rightarrow P | y \rightarrow Q)$
  – where $\alpha(x \rightarrow P | y \rightarrow Q) = \alpha P$
  – provided $(x,y) \subseteq \alpha P$ and $\alpha P = \alpha Q$

• Examples
  – Possible counter moves
    • $(\text{up} \rightarrow \text{STOP} | \text{right} \rightarrow \text{right} \rightarrow \text{up} \rightarrow \text{STOP})$
  – Machine with two combination of changes
    • $CH5C = \text{in5p} \rightarrow (\text{out1p} \rightarrow \text{out1p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow CH5C)$
      | $\text{out2p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow CH5C)$

Choice Example

• Chocolate / toffee machine
  – $VMCT = \mu X: \text{coin} \rightarrow (\text{choc} \rightarrow X | \text{toffee} \rightarrow X)$

• Machine with choices of coins / goods / change
  – $VM = (\text{in2p} \rightarrow (\text{large} \rightarrow VM | \text{small} \rightarrow \text{out1p} \rightarrow VM)$
    | $\text{in1p} \rightarrow (\text{small} \rightarrow VM | \text{in1p} \rightarrow (\text{large} \rightarrow VM | \text{in1p} \rightarrow \text{STOP}))$

• Machine that trusts customers
  – $VMCRED = \mu X: \text{coin} \rightarrow (\text{choc} \rightarrow X | \text{choc} \rightarrow \text{coin} \rightarrow X)$

• To prevent loss, an initial payment is required
  – $VM2 = (\text{coin} \rightarrow VMCRED)$

• Copying Process
  – $COPYBIT = \mu X: (\text{in0} \rightarrow \text{out0} \rightarrow X | \text{in1} \rightarrow \text{out1} \rightarrow X)$
Choice

- More than two alternatives
  - \( (x \rightarrow P | y \rightarrow Q | \ldots | z \rightarrow R) \)
  - where \( x, y, z \) are distinct events

- Incorrect syntax
  - \( (x \rightarrow P | x \rightarrow Q) \)
  - \( (x \rightarrow P | y \rightarrow Q | z \rightarrow R) \)

- In general, choice is written as
  - \( (x : B | P(x)) \)
  - Offers a choice of any event \( x \) in \( B \), and behaves like \( P(x) \)
  - A set \( B \) is called the "initial menu" of the process

- Example: Process which can engage in any event in alphabet \( A \)
  - \( \alpha \text{RUN}_A = A \)
  - \( \text{RUN}_A = (x : A \rightarrow \text{RUN}_A) \)

- There are other kinds of choice: e.g., non-deterministic choice

Mutual Recursion

- Orange-Lemon drink dispenser (\( DD \))
  - Pressing two buttons: \( \text{setorange}, \text{setlemon} \)
  - Dispensing drinks: \( \text{orange}, \text{lemon} \)
  - \( \alpha DD = \alpha O = \alpha L = \{ \text{setorange, setlemon, orange, lemon} \} \)

  \[ DD = (\text{setorange} \rightarrow O \mid \text{setlemon} \rightarrow L) \]

  \[ O = (\text{orange} \rightarrow O \mid \text{setlemon} \rightarrow L \mid \text{setorange} \rightarrow O) \]

  \[ L = (\text{lemon} \rightarrow L \mid \text{setorange} \rightarrow O \mid \text{setlemon} \rightarrow L) \]

- Object movement
  - On the ground: move up or around
    - \( CT_0 = (\text{up} \mid \text{CT}_1 \mid \text{around} \mid CT_0) \)
  - In the air: move up or down
    - \( CT_{n+1} = (\text{up} \mid CT_{n+2} \mid \text{down} \mid CT_n) \)

Laws

- Law1:
  - \( (x \rightarrow P \mid y \rightarrow Q) = (y \rightarrow Q \mid x \rightarrow P) \)
  - \( (x \rightarrow P) \neq \text{STOP} \)
  - \( (x \rightarrow P) \neq (y \rightarrow Q) \) if \( x \neq y \)
  - \( (x \rightarrow P) = (x \rightarrow Q) \) implies \( P = Q \)

- Example:
  - \( (\text{coin} \rightarrow \text{choc} \rightarrow \text{coin} \rightarrow \text{choc} \rightarrow \text{STOP}) \neq (\text{coin} \rightarrow \text{STOP}) \)
  - \( \mu X \cdot (\text{coin} \rightarrow (\text{choc} \rightarrow X \mid \text{toffee} \rightarrow X)) \)
    \[ = \mu X \cdot (\text{coin} \rightarrow (\text{toffee} \rightarrow X \mid \text{choc} \rightarrow X)) \]
Laws

- **Law 2**: let $F(X)$ be a guarded expression
  - $(Y = F(Y)) = (Y = \mu X \cdot F(X))$
  - $\mu X \cdot F(X) = F(\mu X \cdot F(X))$

  - Example:
    - Let $VM1 = (\text{coin} \rightarrow VM2)$
      $VM2 = (\text{choc} \rightarrow VM1)$
    - Then $VM1 = (\text{coin} \rightarrow VM2)$
      $= (\text{coin} \rightarrow (\text{choc} \rightarrow VM1))$
    - $= VMC$

Traces

- **Trace of a behavior of a process**
  - Finite sequence of symbols recording events up to some moment in time

  - Example
    - Chocolate vending machine after serving 2 customers
      - $(\text{coin}, \text{choc}, \text{coin}, \text{choc})$
    - Before anyone puts coins in
      - $(\text{()})$ (called empty trace, the shortest possible trace)
    - Change-giving machine — customer is waiting for the last 2p
      - $(\text{in}5p, \text{out}2p, \text{in}1p)$

Operations on Traces

- **Concatenation**: $s^t$
  - $(\text{coin}, \text{choc})^t = (\text{coin}, \text{choc}, \text{toffee})$

- **Head**: $s_0$
  - $(x, y, x)_0 = x$
  - $(x, y, x)^0 = (y, x)$

- **Tail**: $s'$
  - $(x, y, x)_s = x$
  - $(x, y, x)^t = (y, x)$

- **Ordering**: $s \leq t = (\exists u \cdot s^u = t)$
  - "$s$ is a prefix of $t$"
  - $(x, y) \leq (x, y, x, w)$

- **Star**: $A^* = \{ t \mid t = \langle \rangle \text{ or } (b_0 \in A \text{ and } t' \in A^*) \}$

Operation on Traces

- **After**: $P / s$ where $s \in \text{traces}(P)$
  - "P after s"
  - $(VMC / (\text{coin}) = (\text{choc} \rightarrow VMC)$
  - $(VMC / (\text{coin}, \text{choc})) = VMC$

- **Restriction**: $(t \uparrow A)$ "trace $t$ when restricted to symbols in $A$"
  - $(\text{around}, \text{up}, \text{down}, \text{around}) \uparrow \{ \text{up, down} \} = (\text{up, down})$

- **Length**: $\# t$
  - $\# (x, y, x) = 3$
Traces of Process

• Complete set of all possible traces of $P$
  – $\text{traces}(P)$

• Example
  – $\text{traces(STOP)} = \{ \phi \}$
  – $\text{traces(coin \rightarrow STOP)} = \{ \phi, \langle \text{coin} \rangle \}$
  – $\text{traces(μX \cdot \text{tick} \rightarrow X)} = \{ \phi, \langle \text{tick} \rangle, \langle \text{tick,tick} \rangle, \ldots \} = \{ \text{tick} \}^*$
  – Chocolate vending machine
    • trace($μX \cdot \text{coin} \rightarrow \text{choc} \rightarrow X) = \{ s \mid \exists n \cdot s \leq \langle \text{coin,choc} \rangle^n \}$

Specifications

• Describe intended behaviors of products
  – Assume $tr$ is a variable for an arbitrary trace
  – Let $(tr \downarrow \text{choc})$ denote $#(tr \uparrow \{ \text{choc} \})$
  • Number choc events in trace $tr$

• Example
  – VM owner: # of chocolate must never exceed # of coins inserted
    • $\text{NOLOSS } = (tr \downarrow \text{choc}) \leq (tr \downarrow \text{coin})$
  – VM customers: VM won’t take more coins until it dispenses paid chocolate
    • $\text{FAIR } = (tr \downarrow \text{coin}) \leq (tr \downarrow \text{choc}) + 1$
  – Hence, VM manufacturer must meet spec. from both VM owner and customers
    • $\text{VMSPEC } = \text{NOLOSS } \land \text{FAIR}$
    • $0 \leq (tr \downarrow \text{coin}) - (tr \downarrow \text{choc}) < 1$

Satisfaction

• $P$ sat $S$
  – If a product $P$ meets a specification $S$, then $P$ satisfies $S$
  – Formally, $P$ sat $S$ if $\forall tr \cdot tr \in \text{traces}(P) \Rightarrow S$

• Example: $\text{VMC}$ sat $\text{VMSPEC}$
  – Recall:
    • $\text{VMC} = (\text{coin} \rightarrow (\text{choc} \rightarrow \text{VMC}))$
    • $\text{VMSPEC } = \text{NOLOSS } \land \text{FAIR}$
    • $0 \leq (tr \downarrow \text{coin}) - (tr \downarrow \text{choc}) \leq 1$

Concurrency

• 2 or more processes operating together
  – Syntax: $P \parallel Q$
  – Example: Both customers and vending machines can be viewed as processes interacting with one another

• Interaction
  – Interact via shared events between processes

• Concurrency
  – Specifies how shared and private events in processes are joined
Interaction Example

- A greedy customer tries to get chocolate or toffee without paying.
- If it doesn’t work, he reluctantly pays for chocolate.
  - \( \text{GRCUST} = \{\text{toffee} \rightarrow \text{GRCUST}, \ \text{choc} \rightarrow \text{GRCUST}, \ \text{coin} \rightarrow \text{choc} \rightarrow \text{GRCUST}\} \)
- When using VMCT machine,
  - \( \text{VMCT} = \mu X \cdot (\text{coin} \rightarrow X \mid \text{toffee} \rightarrow X) \)
  - He can’t get goods without paying, hence, he only gets chocolate.
  - \( (\text{GRCUST} \mid \text{VMCT}) = \mu X \cdot (\text{coin} \rightarrow \text{choc} \rightarrow X) \)

Interaction Laws and Traces

- Laws:
  - \( P \parallel Q = Q \parallel P \)
  - \( P \parallel (Q \parallel P) = (P \parallel Q) \parallel P \)
  - \( P \parallel \text{STOP} = \text{STOP} \parallel P \)
  - \( (c \rightarrow P) \parallel (c \rightarrow Q) = (c \rightarrow (P \parallel Q)) \)
  - \( (c \rightarrow P) \parallel (d \rightarrow Q) = \text{STOP} \text{ if } c \neq d \)
- Traces:
  - \( \text{traces}(P \parallel Q) = \text{traces}(P) \cap \text{traces}(Q) \)
  - \( (P \parallel Q) / s = (P / s) \parallel (Q / s) \)

Concurrency

- In general, it’s possible that
  - \( \alpha P \neq \alpha Q \)
  - For \( x \in (\alpha P - \alpha Q) \),
    - \( P \) may engage alone with no concern to \( Q \)
    - And vice versa
- Hence,
  - \( \alpha(P \parallel Q) = \alpha P \cup \alpha Q \)

Concurrency Example

- Noisy vending machine
  - \( \alpha\text{NOISYVM} = \{\text{coin, choc, clink, clunk, toffee}\} \)
  - \( \text{NOISYVM} = (\text{coin} \rightarrow \text{clink} \rightarrow \text{choc} \rightarrow \text{clunk} \rightarrow \text{NOISYVM}) \)
- Customer who cursed when gets chocolate instead of toffee
  - \( \alpha\text{CUST} = \{\text{coin, choc, curse, toffee}\} \)
  - \( \text{CUST} = (\text{toffee} \rightarrow \text{CUST} \parallel \text{curse} \rightarrow \text{choc} \rightarrow \text{CUST}) \)
- Cursing customer using noisy machine
  - \( (\text{NOISY} \parallel \text{CUST}) = \mu X \cdot (\text{coin} \rightarrow (\text{clink} \rightarrow \text{curse} \rightarrow \text{choc} \rightarrow \text{clunk} \rightarrow X \mid \text{curse} \rightarrow \text{clink} \rightarrow \text{choc} \rightarrow \text{clunk} \rightarrow X)) \)
Concurrency Law

- $||$ is symmetric and associative
- $P || STOP_{UP} = STOP_{UP}||P$
- $P || RUN_{UP} = P$

- Let
  - $a \in (\alpha P - \alpha Q)$
  - $b \in (\alpha Q - \alpha P)$
  - $(c,d) \subseteq (\alpha P \cap \alpha Q)$

- Then
  - $(c \rightarrow P) || (c \rightarrow Q) = c \rightarrow (P || Q)$
  - $(c \rightarrow P) || (d \rightarrow Q) = \text{STOP if } c = d$
  - $(a \rightarrow P) || (c \rightarrow Q) = a \rightarrow (P || (c \rightarrow Q))$
  - $(c \rightarrow P) || (b \rightarrow Q) = b \rightarrow ((c \rightarrow P) || Q)$
  - $(a \rightarrow P) || (b \rightarrow Q) = (a \rightarrow (P || (b \rightarrow Q))) \cup (b \rightarrow ((a \rightarrow P) || Q))$

Concurrency Traces

- traces($P || Q$) is all possible traces of process ($P || Q$)

- traces($P || Q$) =
  - $\{ t | (t \uparrow \alpha P) \in \text{traces}(P) \land (t \uparrow \alpha Q) \in \text{traces}(Q) \land t \in (\alpha P \cup \alpha Q)^* \}$

- $t_1 = \langle \text{coin, clink, curse} \rangle \in \text{traces(NOISYVM || CUST)}$
  - $t_1 \uparrow \alpha NOISYVM = \langle \text{coin, clink} \rangle \in \text{traces(NOISYVM)}$
  - $t_1 \uparrow \alpha CUST = \langle \text{coin, curse} \rangle \in \text{traces(CUST)}$

- Similar for $t_2 = \langle \text{coin, curse, clink} \rangle$

Concurrency Pictures
Dining Philosophers

- Book p. 57

Change of Symbol

- Define groups of processes with similar behaviors
- One-on-one function (injection) maps alphabet of P onto a set of symbols A
- \( f : \alpha P \rightarrow A \)

- Example:
  - Recall: Chocolate / toffee machine
    - \( VMCT \rightarrow coin \rightarrow (choc \rightarrow VMCT \rightarrow toffee \rightarrow VMCT) \)
  - Want: Gum / pretzel machine
    - \( f(coin) = coin \)
    - \( f(choc) = gum \)
    - \( f(toffee) = pretzel \)
  - Now, \( VMGP = f(VMCT) \)

Process Labeling

- Label identical but independent processes

- Example:
  - Two machine standing side by side
    - \( (left : VMC) || (right : VMC) \)
  - Possible trace
    - \( (left.coin, right.coin, right.choc) \)

Specifications and Satisfaction

- Satisfaction for concurrent processes

- If \( P \) sat \( S(tr) \) and \( Q \) sat \( T(tr) \) then \( (P \parallel Q) \) sat \( S(tr \uparrow \alpha P) \land T(tr \uparrow \alpha Q) \)
Spec, Sat Example

- Let $\alpha P = \{a, c\}$ and $\alpha Q = \{b, c\}$
  - $P = \{a \rightarrow c \rightarrow P\}$
  - $Q = \{c \rightarrow b \rightarrow Q\}$
- Want to know if $(P || Q)$ sat $0 \leq (\text{tr} \downarrow a) - (\text{tr} \downarrow b) \leq 2$

- Obviously,
  - $P$ sat $0 \leq (\text{tr} \downarrow a) - (\text{tr} \downarrow c) \leq 1$
  - $Q$ sat $0 \leq (\text{tr} \downarrow c) - (\text{tr} \downarrow b) \leq 1$
- Hence,
  - $(P || Q)$ sat $0 \leq (\text{tr} \downarrow a) - (\text{tr} \downarrow c) \leq 1 \\
 0 \leq (\text{tr} \downarrow c) - (\text{tr} \downarrow b) \leq 1$
  - Since $(\text{tr} \uparrow A) ; a = \text{tr} \uparrow a$ when $a \in A$
  - $(P || Q)$ sat $0 \leq (\text{tr} \downarrow a) - (\text{tr} \downarrow c) \leq 1 \\
 0 \leq (\text{tr} \downarrow c) - (\text{tr} \downarrow b) \leq 1$
  - $(P || Q)$ sat $0 \leq (\text{tr} \downarrow a) - (\text{tr} \downarrow b) \leq 2$

Theory of Deterministic Processes

- CSP Book P. 72 – 79
- It shows that
  - CSP laws are in fact true
  - A recursively defined process is indeed a solution of the corresponding recursive equation (fixed point theory)
  - There exists a unique solution