The hierarchy of analytic semantics of weakly consistent parallelism

Jade Alglave (MSR-Cambridge, UCL, UK)
Patrick Cousot (NYU, Emer. ENS, PSL)

REPS AT SIXTY (co-located with SAS 2016)
Edinburgh, Scotland
September 11, 2016
Analytic semantics
Weak consistency models (WCM)

- **Sequential consistency:**
  reads \( r(p, x) \) are *implicitly coordinated* with writes \( w(q, x) \)

- **WCM:**
  *No implicit coordination* (depends on architecture, program dependencies, and explicit fences)
Analytic semantic specification

- **Analytic semantics** = **Anarchic semantics** ∩ **Communication semantics**

- **Anarchic semantics** \(S^a[P]\) :
  describes computations of program \(P\), no constraints on communications

- **Communication semantics**:
  imposes architecture-dependent communication constraints
  
  e.g.: **cat** language (Jade Alglave & Luc Maranget)

---

Hierarchy of anarchic semantics

• Many different styles to describe the same computations e.g.
  • stateless/stateful
  • eager/lazy communications
  • interleaved versus true parallelism
  • …

• They form a hierarchy of abstractions

• The communication semantics is the same for all semantics in the hierarchy
Example: load buffer (LB)

- **Program:**
  \[
  \{ \ x = 0; \ y = 0; \ \}
  
  \begin{align*}
  P_0 & \parallel P_1; \\
  r[] & \ r1 \ x \parallel r[] \ r2 \ y; \\
  w[] & \ y \ 1 \parallel w[] \ x \ 1;
  \end{align*}
  \]

  \[\exists_0: r1=1 \land 1: r2=1\]

- **Example of execution trace** \(t \in S^+ [P]\): 

  \[
  t = w(\text{start}, x, 0) \ w(\text{start}, y, 0) \ r(P0, x, 1) \ \rf[w(P1, x, 1), r(P0, x, 1)] \ w(P0, y, 1) \ r(P1, y, 1) \\
  w(P1, x, 1) \ \rf[w(P0, y, 1), r(P1, y, 1)]
  \]

- **Abstraction to cat candidate execution** \(\alpha_\Xi(t)\):

  \[\begin{array}{c}
  P_0 \quad P_1 \\
  a: \ Rx=1 \\
  b: \ Wy=1 \\
  \parallel \quad \rf \rf \quad \rf \\
  c: \ Ry=1 \\
  d: \ Wx=1
  \end{array}\]
Example: load buffer (LB), cont’d

• **cat specification:**

\[
\text{acyclic (po | rf)}^+ 
\]

The **cat** semantics rejects this execution \( \alpha_{\Xi}(t) : \)

\[
\llbracket \text{cat} \rrbracket (\alpha_{\Xi}(t)) = \text{forbidden}
\]

- **P₀**
  - a: \( Rx=1 \)
  - b: \( Wy=1 \)
- **P₁**
  - c: \( Ry=1 \)
  - d: \( Wx=1 \)

• **The herd7 simulation tool:** [virginia.cs.ucl.ac.uk/herd/](http://virginia.cs.ucl.ac.uk/herd/)
Execution environment

• In general, the semantics of a parallel program depends on hypotheses on the **execution environment**

• e.g.: **coherence order**:

```plaintext
w[[]]x 1; w[[]]x 2; w[[]]x 3; w[[]]x 4;...
```

• the hypotheses on the execution environment (e.g. \( \text{co} \subseteq \text{po} \)) are part of the communication semantics
The WCM semantics

- Abstraction to a candidate execution:

\[
\alpha_\Xi(t) \triangleq \langle \alpha_e(t), \alpha_{po}(t), \alpha_{rf}(t), \alpha_{iw}(t) \rangle \\
\alpha_\Xi(S) \triangleq \{ \langle t, \alpha_\Xi(t) \rangle \mid t \in S \}
\]

- \(\alpha_e(t)\) set of all events
- \(\alpha_{po}(t)\) execution order of events on the same process
- \(\alpha_{rf}(t)\) which read events read from which write events
- \(\alpha_{iw}(t)\) initial write events (initialization before starting the parallel execution)
The WCM semantics

• The cat communication semantics

\[ \mathcal{W}\mathcal{O}\left[ \text{cat} \right] \left( \Xi \right) \]

returns:

• Relations \( \Gamma \) on events representing hypotheses on the execution environment (e.g. co)

• allowed/forbidden depending on whether the candidate execution \( \langle \Xi, \Gamma \rangle \) is consistent or not

\[ \alpha_{\mathcal{W}\mathcal{O}} \left[ \text{cat} \right] (C) \triangleq \{ t, \Gamma \mid \langle t, \Xi \rangle \in C \land \langle \text{allowed}, \Gamma \rangle \in \mathcal{W}\mathcal{O}\left[ \text{cat} \right] (\Xi) \} \]
The WCM semantics

- The WCM semantics:

\[ S[P] \triangleq \alpha_{\text{\textsc{cat}}} \circ \alpha_{\Xi}(S^a[P]) \]

where:

- abstraction to a candidate execution:

\[ \alpha_{\Xi}(S) \triangleq \{ \langle t, \alpha_{\Xi}(t) \rangle \mid t \in S \} \]

- the \textsc{cat} communication semantics:

\[ \alpha_{\text{\textsc{cat}}}(C) \triangleq \{ t, \Gamma \mid \langle t, \Xi \rangle \in C \land \langle \text{allowed}, \Gamma \rangle \in \Xi[\text{\textsc{cat}}](\Xi) \} \]

- The composition of Galois connections.
Definition of the anarchic semantics
Axiomatic parameterized definition of the anarchic semantics

- The semantics $S^\perp[P]$ is a finite/infinite sequence of interleaved events of processes satisfying well-formedness conditions.

- Events:
  - local computations and tests on registers, fences, rmw
  - start writing a shared variable $w(q, x)$
  - start reading of shared variable $r(p, x)$
  - communication event $\text{rf}(w(q, x), r(p, x))$
Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics $S$:
  - uniqueness of events
    \[
    \forall t \in S \cdot \forall t_1, t_2 \in E^*, t_3 \in E^{\infty} \cdot \forall e, e' \in E. (t = t_1 e t_2 e' t_3) \implies (e \neq e'). \quad (Wf_1(S))
    \]
  - traces start with an initialization of the shared variables
    \[
    t = \begin{align*}
      w(\text{start}, x, 0) & \cdot w(\text{start}, y, 0) \\
      r(\text{P0}, x, 1) & \cdot w[P1, x, 1), r[P0, x, 1]] \\
      w(\text{P1}, x, 1) & \cdot r[P1, y, 1] \\
      w(\text{P1}, x, 1) & \cdot r[P0, y, 1), r[P1, y, 1]]
    \end{align*}
    \]
Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics $S$:
  - finite traces are maximal

\[
\forall t \in S \cap \mathcal{E}^+ . \nexists t' \in \mathcal{E}^\infty . t t' \in S .
\] (Wf$_3(S)$)
Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics $S$:

  - **read events must be satisfied by a unique communication event**

\[
\forall t \in S . \forall t_1 \in C^*, t_2 \in C^{*\infty} . (t = t_1 r(p, x) t_2) \implies \\
(\exists t_3 \in C^*, t_4 \in C^{*\infty} . t = t_3 rf[w(q, x), r(p, x)] t_4).
\]

\[
\forall t \in S . \forall t_1, t_2 \in C^*, t_3 \in C^{*\infty} . \\
(t \neq t_1 rf[w(q, x), r(p, x)] t_2 rf[w'(q', x), r(p, x)] t_3).
\]

(Wf$_4(S)$)

(Wf$_5(S)$)
Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics $S$:

  - communications cannot be spontaneous (must be originated by a read and a write)

\[
\forall t \in S. \forall t_1 \in C^*, t_2 \in C^{*\infty}. (t = t_1 \forall [w(q, x), r(p, x)] t_2) \implies
(\exists t_3 \in C^*, t_4 \in C^{*\infty}. t = t_3 w(q, x) t_4 \land \exists t_5 \in C^*, t_6 \in C^{*\infty}. t = t_5 r(p, x) t_6).
\]  

(Wf$_6$(S))
Axiomatic parameterized definition of the anarchic semantics

- The **language**: 

- Programs: `initialisation [P₁ || ⋯ || Pₙ]`

- Actions (labelled `ℓ ∈ ℒ(p)`): 

  - `a ::= m` imperative actions marker
  - `r := e` assignment
  - `r := x` read of shared variable `x`
  - `x := e` write of shared variable `x`
  - `b | ¬b` conditional actions test

- Next action: `next(p, ℓ) nextt(p, ℓ) nextf(p, ℓ)` for tests
Axiomatic parameterized definition of the anarchic semantics

- Example of language-dependent well-formedness condition: computation (markers: skip, fence, begin/end of rmw)

\[
\forall p \in \Pi . \forall k \in [1, 1 + |\tau|] . \forall \ell \in \mathbb{L}(p) . \\
(\exists \theta \in \mathcal{P}(p) . \bar{\tau}_k = m(\langle p, \ell, m, \theta \rangle)) \\
\implies (\ell \in N^p(\tau, k) \land \text{action}(p, \ell) = m) .
\]

(unique) event stamp \( \theta \)

control of process \( p \) is at label \( \ell \)

action of process \( p \) is at label \( \ell \) is the marker action \( m \)
Axiomatic parameterized definition of the anarchic semantics

- Example of language-dependent well-formedness condition: computational (local variable assignment)

\[ (\forall p \in \mathcal{P} . \forall k \in ]1, 1 + |\tau|[ . \forall \ell \in \mathbb{L}(p) . \forall v \in \mathcal{D} . \] 

\[ (\exists \theta \in \mathcal{P}(p) . \overline{\tau}_k = a(\langle p, \ell, r := e, \theta \rangle, v)) \]

\[ \implies (\ell \in N^p(\tau, k) \land \text{action}(p, \ell) = r := e \land v = E^p[e](\tau, k - 1)) . \]

- control of process \( p \) is at label \( \ell \)
- action of process \( p \) is at label \( \ell \) is a register assignment
- value \( v \) of \( e \) is evaluated by past-travel
Media variables

• With WCM there is no notion of “the current value of shared variable $x$”

• At a given time each process may read a different value of the shared variable $x$ (maybe guessed or unknown since a read may read from a future write)

• We use *pythia variables* (to record the values communicated between a write and read, whether the two accesses are on the same process or not)
Axiomatic parameterized definition of the anarchic semantics

- Example: communication

- a read event is initiated by a read action:
  \[\forall p \in \mathbb{P} . \forall k \in ]1, 1 + |\tau|[ . \forall \ell \in \mathbb{L}(p) . \]
  \[(\exists \theta \in \mathcal{P}(p) . (\bar{\tau}_k = \tau(\langle p, \ell, r := x, \theta \rangle, x_\theta))) \]
  \[\implies (\ell \in N^p(\tau, k) \land \text{action}(p, \ell) = r := x) .\]

- a read must read-from (rf) a write (weak fairness):
  \[\forall p \in \mathbb{P} . \forall i \in ]1, 1 + |\tau|[ . \forall r \in \mathcal{W}(p) . \]
  \[\bar{\tau}_i = r \implies (\exists j \in ]1, 1 + |\tau|[ . \exists w \in \mathcal{W} . \bar{\tau}_j = \text{rf}[w, r]) .\]
Axiomatic parameterized definition of the anarchic semantics

- **Predictive evaluation** of pythia variables:

\[ V^0_{(32)}[x_\theta](\tau, k) \triangleq v \text{ where } \exists! i \in [1, 1 + |\tau|]. (\tau_i = v(\langle p, \ell, r := x, \theta \rangle, x_\theta)) \land \exists! j \in [1, 1 + |\tau|]. (\tau_j = r[\nu(\langle p', \ell', x := e', \theta' \rangle, v), \tau_i]) \]

- **Local past-travel** evaluation of an expression:

\[ E^0_{(30)}[r](\tau, k) \triangleq v \text{ if } k > 1 \land \left( (\tau_k = a(\langle p, \ell, r := e, \theta \rangle, v)) \lor (\tau_k = v(\langle p, \ell, r := x, \theta \rangle, x_\theta) \land V^0[x_\theta](\tau, k') = v) \right) \]

\[ E^0_{(30)}[r](\tau, 1) \triangleq l[0] \]

\[ E^0_{(30)}[r](\tau, k) \triangleq E^0_{(30)}[r](\tau, k - 1) \text{ i.e. } \tau_1 = \epsilon_{\text{start}} \text{ by } Wf_{15}(\tau) \text{ otherwise.} \]
Abstractions of the anarchic semantics
Abstractions

- **Anarchic semantics:**
  \[ S^<\downarrow[P] \triangleq \lambda \langle B, \text{sat}, D, I, G, V, E, N \rangle \cdot \{ \tau \in \mathcal{S}[P] | \subseteq | Wf_1(\tau) \land \ldots \land Wf_{29}(\tau) \} \]

- **Examples of abstractions:**
  - Choose data (e.g. ground values, uninterpreted symbolic expressions, interpreted symbolic expressions i.e. “symbolic guess”)
  - Bind parameters (e.g. how expressions are evaluated)
  - …
Binding a parameter of the semantics

- The abstraction

\[ \alpha_a(f) \overset{\text{def}}{=} f(a) \]

\[ \langle \wp(A, B, \ldots) \rightarrow \wp(R), \subseteq \rangle \overset{\alpha_a}{\leftrightarrow} \langle \wp(B, \ldots) \rightarrow \wp(R), \subseteq \rangle \]
The hierarchy of interleaved semantics

WCM

\[ \alpha_\text{cat} \circ \alpha_{\Xi} (S^\perp [P]) \]

valued

symbolic interpreted

symbolic uninterpreted

data generic

locally sequential

unspecified locality

inscrutable

unspecified predictability

predictive

\[ \alpha_\text{cat} \circ \alpha_{\Xi} \]

\[ S^{vi} [P] = S^{vo} [P] \]

\[ \alpha_u \]

\[ \alpha_{vi} \]

\[ \alpha_{oi} \]

\[ \alpha_{ui} \]

\[ \alpha_u \]

\[ \alpha_\text{sat} \]

\[ \alpha_{V(34)} \]

\[ \alpha_{V(32)} \]

\[ \alpha_{E(30), N(31)} \]

\[ \alpha_\text{sat} \]

\[ \alpha_{D, li} \]

\[ \alpha_{B, sat, D, li} \]

Fig. 5. Hierarchy of time-travel, stateless, maximal, interleaved, stepless trace semantics

ACM Transactions on Programming Languages and Systems, Vol. V, No. N, Article A, Publication date: January YYYY.
True parallelism with local communications

- Extract from interleaved executions:
  - The subtrace of each process keeping communications in the process that read

\[\Rightarrow\] no more global time between processes

\[\Rightarrow\] local time between local actions and communications (a read can still tell when it is satisfied by which write)
True parallelism with local communications

- **Interleaved execution:**
  \[
  t = \begin{align*}
  w(\text{start}, x, 0) & \quad w(\text{start}, y, 0) \\
  r(P0, x, 1) & \quad rf[w(P1, x, 1), r(P0, x, 1)] \\
  w(P1, x, 1) & \quad rf[w(P0, y, 1), r(P1, y, 1)] \\
  \end{align*}
  \]

- **Parallel executions with interleaved communications:**
  \[
  t = \text{Initialization:} \quad w(\text{start}, x, 0) \quad w(\text{start}, y, 0)
  \]
  \[
  \begin{align*}
  P0: & \quad r(P0, x, 1) \quad rf[w(P1, x, 1), r(P0, x, 1)] \quad w(P0, y, 1) \\
  P1: & \quad r(P1, y, 1) \quad w(P1, x, 1) \quad rf[w(P0, y, 1), r(P1, y, 1)]
  \end{align*}
  \]
True parallelism of computations and communications

- Extract from interleaved executions:
  - The *subtrace of each process* (sequential execution of actions)
  - The *rf communication relation* (interactions between processes)

  $\Rightarrow$ *no more global time* between processes

  $\Rightarrow$ *no more global/local time* for communications
True parallelism with separate communications

- Parallel executions with interleaved communications:

  **Initialization:** \[ \begin{align*} & w(\text{start}, x, 0) \quad w(\text{start}, y, 0) \\ & r(P_0, x, 1) \quad w(P_0, y, 1) \\ & r(P_1, y, 1) \quad w(P_1, x, 1) \end{align*} \]

  **Communications:** \[ \{ rf[ w(P_1, x, 1), r(P_0, x, 1)], \quad rf[ w(P_0, y, 1), r(P_1, y, 1)] \} \]
True parallelism with separate communications

- This is the semantics used by the herd7 tool:

\[ P_0 \rightarrow \text{event} \rightarrow a: Rx=1 \rightarrow \text{po} \rightarrow b: Wy=1 \rightarrow \text{event} \rightarrow c: Ry=1 \rightarrow \text{po} \rightarrow d: Wx=1 \rightarrow \text{event} \rightarrow P_1 \]

+ interpreted symbolic expressions i.e. “symbolic guess”
The true parallelism hierarchy

![Diagram of the true parallelism hierarchy]

- Separated communication
- True parallelism
- Per process

WCM

- $\alpha^{\text{cat}} \circ \alpha_{\equiv}(S[P])$

Interleaved communication

- Locally sequential

Free, Eager, Lazy
States

• At each point in a trace, the state abstracts the past computation history up to that point

• Example: classical environment (assigning values to register at each point k of the trace):

\[
\rho^p(\tau, k) \triangleq \lambda r \in R(p) \cdot E^p[r](\tau, k)
\]

\[
\nu^p(\tau, k) \triangleq \lambda x_\theta \cdot V^p_{(32)}[x_\theta](\tau, k)
\]
Prefixes, transitions, ...

• Abstract traces by their prefixes:

\[ \overleftarrow{\alpha}(S) \triangleq \bigcup \{ \overleftarrow{\alpha}(\tau) \mid \tau \in S \} \]
\[ \overleftarrow{\alpha}(\tau) \triangleq \{ \tau[\{j\} \mid j \in [1, 1 + |\tau|]\} \]
\[ \tau[\{j\}] \triangleq \langle \overleftarrow{\tau}_i \rightarrow \tau_i \mid i \in [1, 1 + j] \rangle \]

• and transitions: extract transitions from traces

\[ \Rightarrow \text{communication fairness is lost, inexact abstraction,} \]
\[ \Rightarrow \text{add fairness condition} \]
\[ \Rightarrow \text{impossible to implement with a scheduler (≠ process fairness)} \]
Effect of the cat specification on the hierarchy
Exactness and cat preservation

\[ \alpha_{\text{[cat]}} \circ \alpha_{\Xi}(\alpha(S^\perp[P])) = \alpha(\alpha_{\text{[cat]}} \circ \alpha_{\Xi}(S^\perp[P])) \]

WCM semantics

Concurrent execution semantics

Exact and cat-preserving semantics

Anarchic semantics

Semantics

\( \emptyset \)

\( \mathcal{E}^\infty \)
The cat abstraction

- The same cat specification applies equally to any concurrent execution abstraction parallel semantics in the hierarchy.

- The appropriate level of abstraction to specify WCM:
  - No states, only marker (e.g. fence), r, w, rf(w,r) events
  - No values in events
  - No global time (only po order of events per process)
  - Time of communications forgotten (only rf of who communicates with whom)
  - Hypotheses on the execution environment independent of computed and communicated values
Conclusion
Conclusion

- **Analytic semantics**: a new style of semantics

- The hierarchy of *anarchic semantics* describes the same computations and potential communications in very different styles

- The *cat semantics* restricts communications to a machine/network architecture in the same way for all semantics in the hierarchy

- This idea of *parameterized semantics at various levels of abstraction* is useful for
  - Verification
  - Static analysis
The End