A Logical Framework for Self-Optimizing Networked Cyber-Physical Systems

Mark-Oliver Stehr, Minyoung Kim, Carolyn Talcott

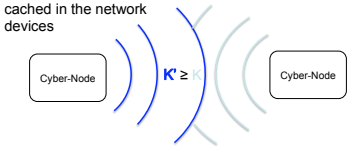
http://ncps.csl.sri.com

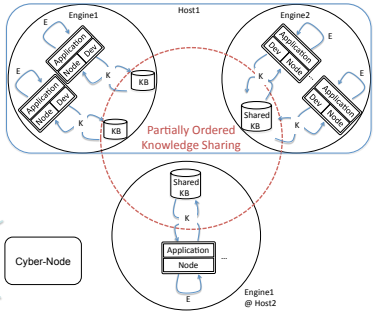
Students: Jinwoo Kim (Seoul National Univ.), Vincent Wang (UPenn)

SRI International

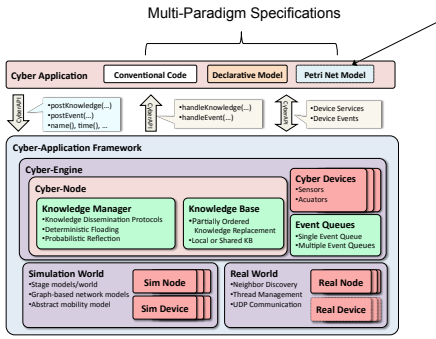
An Application Framework for Networked CPS

- Based on new loosely-coupled distributed computing model:
  Partially Ordered Knowledge Sharing
- Inspired by our earlier work on delay-/disruption-tolerant networking (DTN)
- Minimal assumptions on network connectivity (can be very unreliable)
- Works with dynamic topologies, network partitions, and mobile nodes
- Designed for heterogeneous networking technologies and heterogeneous nodes
- Partial order allows the network to replace obsolete or subsumed knowledge
- Global consistency is not enforced (impossible in challenging environments)
- Avoids strong non-implementable primitives, e.g. transactions
- Locally each cyber-node uses an event-based model with local time
- Events are local, but knowledge can be shared and cached in the network
- Each cyber-node can have attached cyber-physical devices
- Framework supports
  - model-based simulation/analysis mode
  - real-world deployment/execution mode
- Applications cannot distinguish between simulation and reality

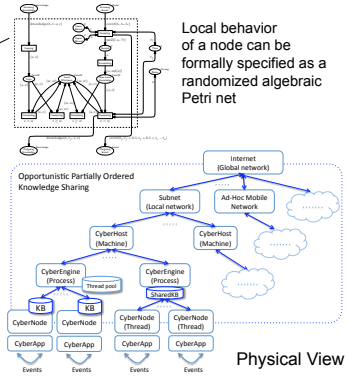




Logical View



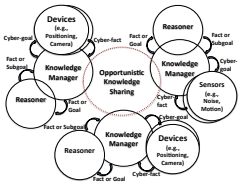
Architectural View



Physical View

Distributed Declarative Control

Logical Theory for Distributed Surveillance by a Team of Mobile Robots in an Instrumented Space



Forward Clauses:

$F1: Noise(T, A) \Rightarrow Trigger(T, A).$
$F2: Motion(T, A) \Rightarrow Trigger(T, A).$
$F3: Adjacent(A, B) \Rightarrow Adjacent(B, A).$

Backward Clauses:

$B1: Interest(T_I, I, R) \Leftarrow Result(T_I, T_T, 0, I), Deliver(T_I, T_T, 1, I, R).$
$B2: Deliver(T_I, T_T, N_D, I, R) \Leftarrow Delivered(T_I, T_T, N_D, I, R).$
$B3: Deliver(T_I, T_T, N_D, I, R) \Leftarrow$
$\quad Position(T_P, R, A), Position(T'_P, R', A'), R' \neq R,$
$\quad MoveTo(T_I, T_T, N_D, 0, \infty, R', A), Deliver(T_I, T_T, N_D, I, R).$
$B4: Result(T_I, T_T, N_D, I') \Leftarrow CompImage(T_I, T_T, N_D, I), I' = Extract(I).$
$B5: CompImage(T_I, T_T, N_D, I') \Leftarrow RawImage(T_I, T_T, N_D, I), I' = Compress(I).$
$B6: RawImage(T_I, T_T, N_D, I) \Leftarrow Trigger(T_T, A), T_I \leq T_T,$
$\quad MoveTo(T_I, T_T, N_D, 0, T_T + \Delta t_{ad}, R, A),$
$\quad TakeSnapshot(T_I, T_T, N_D, T_T + \Delta t_{sd}, A, I).$
$B7: TakeSnapshot(T_I, T_T, N_D, D, A, I) \Leftarrow$
$\quad Snapshot(T_I, T_T, N_D, T_S, A, I), T_T \leq T_S, T_S \leq D.$
$B8: MoveTo(T_I, T_T, N_D, W', D, R, B) \Leftarrow Position(T_P, R, B), T_P \leq D.$
$B9: MoveTo(T_I, T_T, N_D, W', D, R, B) \Leftarrow Adjacent(A, B), W' > -b_w, W = W' - 1,$
$\quad MoveTo(T_I, T_T, N_D, W, D, R, A), Move(T_I, T_T, N_D, W', D, R, A, B).$

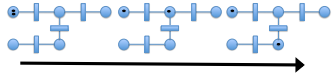
Replacement Ordering: (f denotes a fact and g a goal and x denotes either)

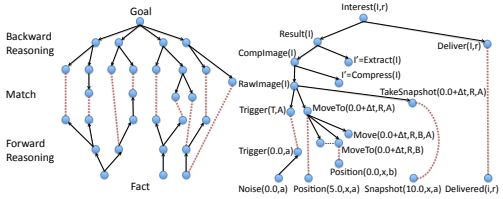
$O1: f: Position(t_P, r, \ldots) \prec f: Position(t'_P, r, \ldots)$ if $t_P < t'_P$.
$O2: x: X(t_I, \ldots) \prec g: Interest(t'_I, \ldots)$ if $t_I < t'_I$.
$O3: x: X(t_I, t_T, n_D, \ldots) \prec f: Result(t_I, t_T, n_D, \ldots)$ if $x: X \neq f: Result.$
$O4: x: X(t_I, t_D, n_D, \ldots) \prec f: Deliver(t_I, t_D, n_D, \ldots)$ if $x: X \neq f: Deliver.$

Variables: T: time, D: snapshot deadline, A and B: area, R: robot, I: image or derived information, N: identifier, W: weight
Constants: Δt_{sd}: relative snapshot deadline (max. delay from trigger event), b_w: bound for weight (diameter of the floor plan)

- Challenges:
  - Traditional logics are not designed for distributed reasoning
  - Logics are traditionally closed systems, i.e. not interactive
- Here we consider the NCPS as a single asset
  - Logical theory/specification is available to all nodes
  - Nodes contribute resources according to their capabilities
- Knowledge = Facts + Goals is transparently shared
  - Facts can represent observations
  - Goals can represent control objectives
- Distributed logical framework
  - Integrated forward and backward reasoning
  - Partial order is essential part of the distributed logic
- Tested with abstract mobility model and Stage multi-robot simulator

Abstract Mobility Model with two Robots

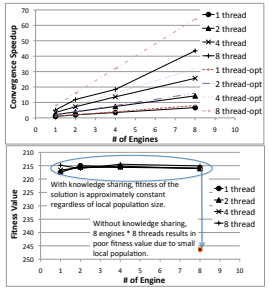


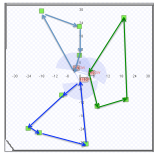


Distributed Dynamic Optimization

- Distributed optimization fits well into the partial-ordered knowledge-sharing model
- Replacement order is defined by objective function (solution fitness)
- Case study: Multi TSP applied to sensor data collection by a team of mobile robots
- Algorithm: Distributed version of a quantum-evolutionary optimizer
- Optimizer performance is studied on Internet-wide testbed (Planet Lab)
- Same optimizer can operate as a cyber-node in an NCPS (e.g. in each robot)
- Optimization problem becomes dynamic (feedback trough environment)
- Key features: robustness and fault-tolerance

Planet Lab Deployment





Possible next steps:
- Generalization: use of a generic optimization framework
- Distributed multi-objective optimization (Pareto optimality)
- Combining optimization and declarative control
- Use of Weighted/Quantitative/Probabilistic Logic

Stage Multi-Robot Simulator

Robot Failure

Opportunistic Knowledge Sharing