Pei (https://blogs.cuit.columbia.edu/zp2130/)

Search

Search

+ Add post (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post-new.php)
Menu

- Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/)
- Posts (https://blogs.cuit.columbia.edu/zp2130/posts/)
- Resources (https://blogs.cuit.columbia.edu/zp2130/resources/)
- Cacti-based Framework (https://blogs.cuit.columbia.edu/zp2130/cacti/)
- Publications (https://blogs.cuit.columbia.edu/zp2130/publications/)

Email Address:

zp2130@caa.columbia.edu (mailto:zp2130@caa.columbia.edu)

p@caa.columbia.edu (mailto:p@caa.columbia.edu)

Blog Stats

137,046 hits

State Action/Control

blogs.cuit.columbia.edu/p (https://blogs.cuit.columbia.edu/p/)

Meta

Site Admin (https://blogs.cuit.columbia.edu/zp2130/wp-admin/)

Log out (https://blogs.cuit.columbia.edu/zp2130/wp-login.php?action=logout&_wpnonce=f713894243)

Entries feed (https://blogs.cuit.columbia.edu/zp2130/feed/)

Comments feed (https://blogs.cuit.columbia.edu/zp2130/comments/feed/)

WordPress.org (https://wordpress.org/)

Decentralized Optimal Control of Distributed Interdependent Automata With Priority Structure

Decentralized Optimal Control of Distributed Interdependent Automata With Priority Structure (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Decentralized-Optimal-Control-of-Distributed-Interdependent-Automata-With-Priority-Structure.pdf)

Data Flowchart (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Decentralized_Optimal_Control_of_Distributed_Interdependent_Automata_With_Priority_Structure-1.pdf)

Notation

$P^i = \left(T, U^i, X^i, Y^i, W^i, f^i, g^i\right)$: **subsystem model**, the **plant** ***P*** ^{i}, deterministic finite-state automaton.

$T = 0, 1, 2, ... \subset \mathbb{N} \cup \{0\}$ with $k \in T$:an event time.

${\nu_k}^i \in U^i = \{1, ..., m_i\} \subset \mathbb{N}$:finite sets of discrete inputs.

${\xi_k}^i \in X^i = \{1, ..., n_i\} \subset \mathbb{N}$:discrete states.

${y_k}^i \in Y^i = \{1, ..., q_i\} \subset \mathbb{N}$:finite sets of discrete outputs.

$W^i \in \{0, 1, ..., q'\}^{m_i \times n_i}$:dependence matrix.$W^i(input, state) = y^{i+1} > 0 \ or \ = 0$

$f^i : X^i \times U^i \rightarrow X^i$:A deterministic state transition function.

$g^i : X \rightarrow Y^i$:output function of Moore-type.

$T = 0, 1, 2, ... \subset \mathbb{N} \cup \{0\}$ with $k \in T$:an event time.

${\xi_0}^i \in X^i$:initial state.

${\nu_k}^i \in U^i \cup \{0\} = \{0, 1, ..., m_i\}$:finite sets of discrete inputs.

${\phi^i}_u := \{\nu^i_0, \nu^i_1, ...\}$:input sequence.

${\phi^i}_x := \{\xi^i_0, \xi^i_1, ...\}$:the elements of an admissible run.

${\phi^i}_y := \{y^i_0, y^i_1, ...\}$:output sequence.

$$\xi^i_{k+1} = \begin{cases} \xi^i_k, & if\ \nu^i_k = 0\ . \\ f^i(\xi^i_k, \nu^i_k), & if\ \nu^i_k \in U^i, \quad W(\nu^i_k, \xi^i_k = 0) \\ f^i(\xi^i_k, \nu^i_k), & if\ \nu^i_k \in U^i, \quad W^i(\nu^i_k, \xi^i_k) \in Y^{i+1} \end{cases} \quad (1)$$

$$y^i_{k+1} = g^i(\xi^i_{k+1}) \quad (2)$$

${x_k}^i \in \{0,1\}^{n_i \times 1}$:state vector.

$$\begin{Bmatrix} {x_k}^i_{,\ j} = 1 \\ {x_k}^i_{,\ p} = 0 \end{Bmatrix}$$ if and only if ${\xi_k}^i = j \in \{1, ..., n_i\}$ is the active state of P^i in k.(3)

${F_l}^i \in \{0,1\}^{n_i \times n_i}$:state transition matrix,for any $input\ l \in U^i$.

$h, j \in X^i$.

$${F_l}^i(j, h) = \begin{cases} 1, & \text{if } j = f^i(h, l) \\ 0, & \text{otherwise.} \end{cases}$$ (4) : ***P^i*** can be transitioned from state ${\xi_k}^i = h$ into state ${\xi_k}^i = j$ if the *input l* is applied.

${F_l}^i(j, j) = 1$ if ${F_l}^i(p, j) = 0\ \forall p \neq j,\ p \in \{1, ..., n_i\}$.

$\sum_{j=1}^{n_i} {F_l}^i(j, h) = 1$ applies $\forall h \in \{1, ..., n_i\}$ and $l \in U^i$.

$\mathfrak{F}^i = \{F^i_1, ..., F^i_{m_i}\}$: *set of state transition matrices.*

$x^i_0 \in \{0,1\}^{n_i \times 1}$: *initial vector.* $\sum_{j=1}^{n_i} x^i_{0,j} = 1$.

${\phi^i}_u := \{\nu^i_0, \nu^i_1, ...\}$:input sequence.

${\phi^i}_x := \{x^i_0, x^i_1, ...\}$:a run of P^i over T is admissible.

$$x^i_{k+1} = \begin{cases} x^i_k, & \text{if } \nu^i_k = 0 \\ {F_j}^i \cdot x^i_k, & \text{if } \nu^i_k \in U^i, \quad W(\nu^i_k, \xi^i_k) = 0 \\ {F_j}^i \cdot x^i_k, & \text{if } \nu^i_k \in U^i, \quad W^i(\nu^i_k, \xi^i_k) \in X^{i+1}. \end{cases} \quad (5)$$

$F^i = \sum_{l=1}^{m_i} {F_l}^i$:accumulated state transition matrix. It encodes with $F^i(j, h) = 1$ that the transition from $\xi^i_k = h$ to ${\xi_k}^i = j$ is possible with at least one input. (6)

$R^i := \sum_{p=1}^{n_i} (F^i)^p$:a reachability matrix. To this one-step from $\xi^i_k = h$ to ${\xi_k}^i = j$ reachability, ***R^i*** formalizes the **possibility** of transferring ***P^i*** between an arbitrary pair of states. (6)$R^i(j, h) = 1$ models that state ***j*** is reachable from state ***h*** by at least one input sequence in at most ***n_i*** state **transitions**.

$\pi(\xi^i_k, \xi^i_{k+1}, \nu^i_k)$:transition costs. for any transition $\xi^i_{k+1} = f^i(\xi^i_k, \nu^i_k)$ specified for ***P^i*** through the state transition function (or the set of state transition matrices $\mathfrak{F}^i$). Possible interpretations of such transition costs are the time, the control effort, and/or the energy required to steer ***P^i*** $from\ \xi^i_k\ to\ {\xi^i}_{k+1}$ by the use of the control input ${\nu^i}_k$, i.e., π can encode state and control costs.

${\Pi_j}^i(q, p) = \pi(p, q, j)$:cost of the transition- $f^i(p, j) = q$.

${\Pi^i}_j(q, p) = \infty$:if the transtion is infeasible for input j. $F^i_j(q, p) = 0$.

${\Pi^i}_j(p, p) = 0$: self-loops, $\forall p \in X^i, j \in U^i$.

$$\Pi^i := \{q, p \in \chi^i : min_{j \in U^i} \Pi^i_j(q, p)\} \quad (7)$$

$f^i(p, j) = q$ over all $j \in U^i$ values.

$$\Pi^i_{U^i} := \left\{q, p \in \chi^i : \Pi^i_{U^i} = 0 \texttt{ if } F^i(p, q) = 0, \text{and}: \Pi^i_{U^i}(q, p) = argmin_{j \in U^i} \Pi^i_j(q, p) \texttt{ if } F^i(p, q) = 1\right\} \quad (8)$$

Π^i_{opt}`:minimal transfer costs for` P^i.

$\Pi^i_{opt}(q, p)$: The minimal costs for transferring the subsystem from the state $\xi^i = p$ `into` $\xi^i = q$.

Problem 1: Subsystem *independent* of other subsystems

$W^i \in \{0, 1, ..., q'\}^{m_i \times n_i}$`:dependence matrix.`

$W^i(input, state) = y^{i+1} = 0^{m_i \times n_i}$`:subsystem` P^i `as independent of other subsystems.`

ξ^i_F`:goal state.`

$\nu^i_k = u^i \cdot K^i \cdot x^i_k \in U^i$.

$u^i = [1, 2, ..., m_i]$`:a row vector of all input indices in` U^i.

$K^i \in \{0, 1\}^{m_i \times n_i}$`:Controller matrix.`

$$\Pi^i_{opt}\left(\xi^i_F, \xi^i_0\right) := min_{\phi^i_u} \sum_{j=1}^{d^i} \Pi_{\nu^i_{j-1}}\left(\xi^i_j, \xi^i_{j-1}\right).$$

```
The task is to compute a state-feedback controller,
```

`which realizes for any arbitrary initialization`$\xi^i_0 \in X^i$`:arbitrary initialization.an input sequence` $\phi^i_u = \{\nu^i_0, \nu^i_1, ..., \nu^i_{d^i-1}\}$`:input sequence.that leads to an admissible run`$\phi^i_\xi = \left(\xi^i_0, ..., \xi^i_{d^i}\right)$`:admissible run.`

1. The final state is the goal $\xi^i_{d^i} = \xi^i_F$
2. The state-feedback control law has the structure: $\nu^i_k = u^i \cdot K^i \cdot x^i_k \in U^i \leftarrow u^i = [1, 2, ..., m_i]$`:a row vector of all input indices in` U^i.$K^i \in \{0, 1\}^{m_i \times n_i}$`:Controller matrix.` (9)
3. The costs of ϕ^i_x are minimal over all admissible runs to transfer ***P^i*** from ξ^i_0 to ξ^i_F : $\Pi^i_{opt}\left(\xi^i_F, \xi^i_0\right) := min_{\phi^i_u} \sum_{j=1}^{d^i} \Pi_{\nu^i_{j-1}}\left(\xi^i_j, \xi^i_{j-1}\right)$.(10)

(9): `For a given` x^i_k`:current state,`$u^i \cdot K^i$`:selects the control input` ν^i_k `to be applied`, in order to **trigger** the next state **transition**. The selection of ***K^i*** according to the solution of (10) produces the ξ^i_F`th row of the matrix` Π^i_{opt}, and establishes an **optimal controller** for ***P^i*** with ξ^i_F`:goal state.`

Synthesis Algorithm for Independent Subsystems

$K^i \in \{0, 1\}^{m_i \times n_i}$`:Controller matrix.`

$$\Pi^i_{opt}\left(\xi^i_F, \xi^i_0\right) := min_{\phi^i_u} \sum_{j=1}^{d^i} \Pi_{\nu^i_{j-1}}\left(\xi^i_j, \xi^i_{j-1}\right).$$

ξ^i_F`th row of the matrix` Π^i_{opt}

ξ^i_F`:goal state.`

Compute **controller matrix**, and the part of **costs** referring to the **goal state**.

Control of Distributed Systems with Linear Structure

$\mathbb{P} = \{P^1, ..., P^i, ..., P^z\}$`:optimal control of processes consisting of z subsystems, interconnected in a linear structure.`

P^i has higher priority than P^{i+1}.

C^i:controller, communicates with $C^{i-1}$and C^{i+1}, $i \in \{2, ..., z-1\}$.

Task of Distributed Controller Synthesis

Assumption 3 : Any subsystem $P^i \in \mathbb{P}, i \in \{2, ..., z\}$ is completely controllable: $R^i = 1^{n_i \times n_i}$.

Proposition 2 : Let $P = \{P^1, P^2\}$ be a pair of 2 connected subsystems for which an admissible run is a sequence of state pair (ξ_k^1, ξ_k^2) according to Definition 2 with $W^1(\nu_k^1, \xi_k^1) \in X^2$ and $W^2 = 0^{m_2 \times n_2}$. The structure is completely controllable if ***P¹*** and ***P²*** on their own are completely controllable according to Assumption 3.

Proof : Since the **transition** of ***P²*** are independent of the **current state** of ***P¹*** , and since subsystem ***P²*** is completely controllable, a sequence ϕ_u^2 of inputs exists to transfer ***P²*** from an arbitrary initial state ξ_0^2 into ξ_F^2.

Thus, ***P²*** is able to deliver any arbitrary output sequence ϕ_y^2 (and thus admissible run ϕ_x^2) to subsystem ***P¹*** , i.e., any condition formulated for ***P¹*** in terms of the dependence matrix W^1 is satisfiable by ***P²*** . Since ***P¹*** itself is completely controllable as well, a sequence of input ϕ_u^1 exists which transfers ***P¹*** into an arbitrary **goal state** ξ_F^1.

Problem 2: Two subsystems

For 2 subsystems P^1 and P^2, let the **goal states** ξ_F^1 and ξ_F^2 be defined. The control task is to compute 2 local feedback control laws, which generate for any initialization $\xi_0^1 \in X^1$ and $\xi_0^2 \in X^2$, the input sequences $\phi_u^1 = \left(\nu_0^1, ..., \nu_{d_1^1-1}^1\right)$ and $\phi_u^2 = \left(\nu_0^2, ..., \nu_{d_2^2-1}^2\right)$, such that the following hold.

1. The admissible runs $\phi_x^1 = \left(\xi_0^1, ..., \xi_{d_1^1}^1\right)$ with $\xi_{d_1^1}^1 = \xi_F^1$ and $\phi_x^2 = \left(\xi_0^2, ..., \xi_{d_2^2}^2\right)$ with $\xi_{d_2^2}^2 = \xi_F^2$.
2. ϕ_u^1 and ϕ_u^2 follow from controllers of the following type: $\nu_k^1 = u^1 \cdot K^1\left(\xi_k^2\right) \cdot x_k^1 \in U^1, \nu_k^2 = u^2 \cdot K^2 \cdot x_k^2 \in U^2$**(11)** with vectors u^1 and u^2, and matrix K^2 as in problem 1, and $K^1\left(\xi_k^2\right) \in \{0,1\}^{m_1 \times n_1}$ for $\xi_k^2 \in X^2$.
3. The global path costs are minimal $J_g = \sum_{k=1}^{d_1^1} \Pi_{\nu_{k-1}^1}\left(\xi_k^1, \xi_{k-1}^1\right) + \sum_{k=1}^{d_2^2} \Pi_{\nu_{k-1}^2}\left(\xi_k^2, \xi_{k-1}^2\right)$**(12)**

Thus, the solution is targeted to provide local controllers C^1 and C^2 for P^1 and P^2, such that the latter are driven from an arbitrarily chosen initial state into the respective local goal state, while the sum of the transfer costs for both control loops is as small as possible.

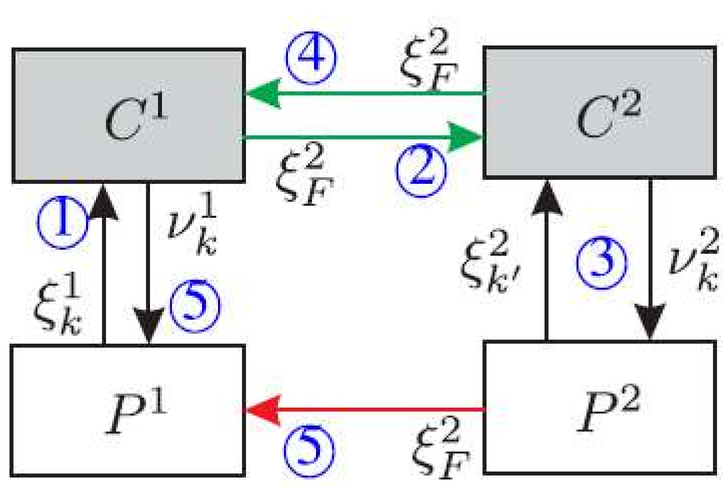

Fig. 4. Online-execution for one state transition of P^1 including the provision of ξ_k^2 by P^2. The numbers indicate the order of information processing.

1. when C^1 receives the information from P^1 that state ξ_k^1 is reached ,

2. C^1 sends the request to C^2 that P^2 has to reach ξ_F^2 as a temporary goal state. This state is encoded in K^1 in order to realize a cost-optimal path of P^1 into its goal state ξ_F^1.

3. Then, C^2 realizes a path of P^2 into ξ_F^2. If the path comprises more than one transition, the pair (P^1,C^1) waits in state ξ_k^1 until P^2 has reached ξ_F^2 ($\xi_{k'}^2$ and $\nu_{k'}^2$ the index **k'** in Fig. 4 is meant to indicate that (P^2,C^2) evolve, while (P^1,C^1) wait in step k).

4. When P^2 attains ξ_F^2, C^2 communicates to C^1 that the requested state is reached .

5. Eventually, the control input ν_k^1 supplied by C^1 together with ξ_F^2 send by P^2 triggers the state transition in P^1 .

Control of Distributed Systems with Tree Structure

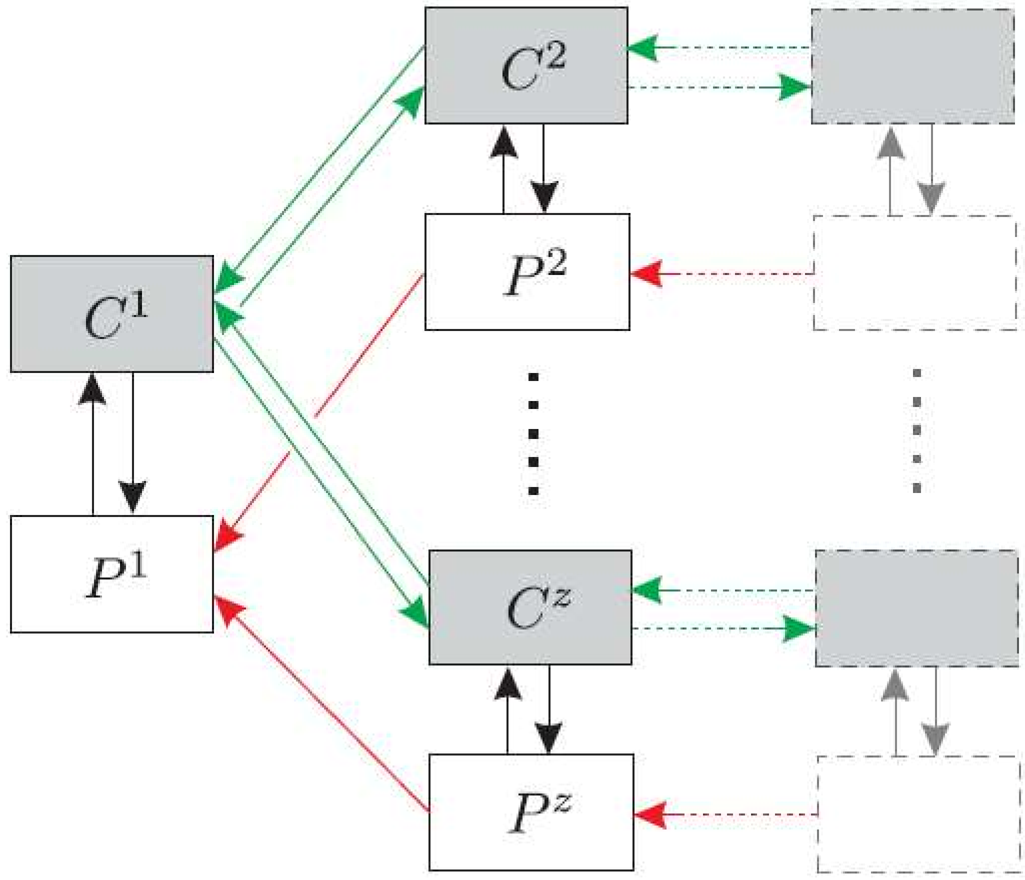

Fig. 7. Distributed system with tree structure where the feedback loop (P^1, C^1) depends on the loops of the subsystems (P^2, C^2) to (P^z, C^z) (each of which may depend on further subordinated subsystems).

edit (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post.php?post=4992&action=edit)

Author: Z Pei (https://blogs.cuit.columbia.edu/zp2130/author/zp2130/) on March 5, 2019

Categories: AI (https://blogs.cuit.columbia.edu/zp2130/category/ai/), Control (https://blogs.cuit.columbia.edu/zp2130/category/control/), Optimal Control (https://blogs.cuit.columbia.edu/zp2130/tag/optimal-control/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/category/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/category/rl/)

Tags: AI (https://blogs.cuit.columbia.edu/zp2130/tag/ai/), Control (https://blogs.cuit.columbia.edu/zp2130/tag/control/), Optimal Control (https://blogs.cuit.columbia.edu/zp2130/tag/optimal-control/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/tag/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/tag/rl/)

Other posts

Neural-network-based decentralized control of continuous-time nonlinear interconnected systems with unknown dynamics (https://blogs.cuit.columbia.edu/zp2130/neural-network-based_decentralized_control_of_continuous-time_nonlinear_interconnected_systems_with_unknown_dynamics/) «» Euler Angles Bunge vs Elastic (https://blogs.cuit.columbia.edu/zp2130/euler_angles_bunge_vs_elastic/)

Last posts

- Symbolic Netlist to Innovus-friendly Netlist (https://blogs.cuit.columbia.edu/zp2130/symbolic_netlist_to_innovus-friendly_netlist/)
- Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms (https://blogs.cuit.columbia.edu/zp2130/finite-sample_convergence_rates_for_q-learning_and_indirect_algorithms/)
- Solving H-horizon, Stationary Markov Decision Problems In Time Proportional To Log(H) (https://blogs.cuit.columbia.edu/zp2130/paul_tseng_1990/)
- Randomized Linear Programming Solves the Discounted Markov Decision Problem In Nearly-Linear (Sometimes Sublinear) Run Time (https://blogs.cuit.columbia.edu/zp2130/randomized_linear_programming_solves_the_discounted_markov_decision_problem_in_nearly-linear_sometimes_sublinear_run_time/)
- KL Divergence (https://blogs.cuit.columbia.edu/zp2130/kl_divergence/)
- The Asymptotic Convergence-Rate of Q-learning (https://blogs.cuit.columbia.edu/zp2130/the_asymptotic_convergence-rate_of_q-learning/)
- Hierarchical Apprenticeship Learning, with Application to Quadruped Locomotion (https://blogs.cuit.columbia.edu/zp2130/hierarchical_apprenticeship_learning_with_application_to_quadruped_locomotion/)
- Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/policy_gradient_methods/)
- Actor-Critic Algorithms for Hierarchical Markov Decision Processes (https://blogs.cuit.columbia.edu/zp2130/actor-critic_algorithms_for_hierarchical_markov_decision_processes/)
- Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation (https://blogs.cuit.columbia.edu/zp2130/hierarchical_deep_reinforcement_learning_integrating_temporal_abstraction_and_intrinsic_motivation/)