Google Research

Resistance Distance and Control Performance for Bittide Synchronization

Sanjay Lall, Google Research and Stanford University
Călin Caşcaval, Martin Izzard, and Tammo Spalink, Google Research

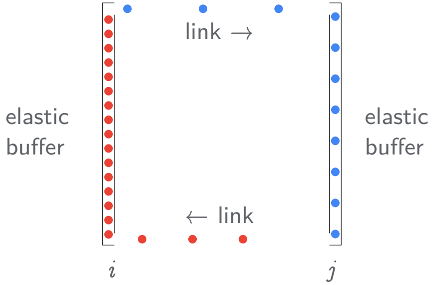
Overview

- *bittide mechanism*
  - wired networks implicitly carry synchronization information
  - mechanism extracts that information and uses it to synchronize machines
  - generates logical synchrony, allowing efficient deterministic execution
- *research project*
  - originated at Princeton in Ph.D. thesis: T. Spalink, *Deterministic sharing of distributed resources*, 2006.
  - now a Google project
  - broad scope: applications, scheduling, simulation, hardware, theory
- *outline:* the mechanism, logical synchrony, controlling frequency

Overview of bittide

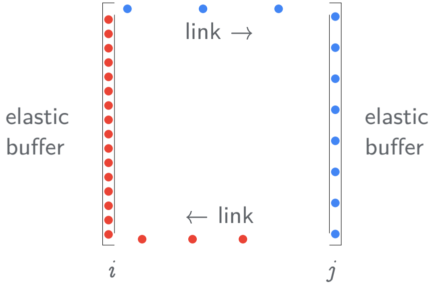
- very low communication overhead
- completely decentralized
- shared *logical* time
  - from the inside, behaves like a system with single shared physical clock
  - from the outside, node clock periods can vary

Overview



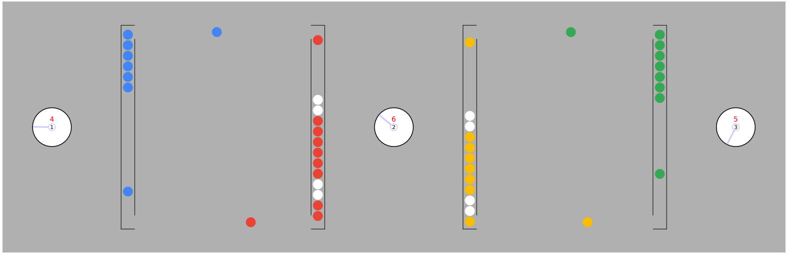
- simplest version of bittide synchronization
- at each node i there is a processor and clock
- nodes are directly connected to neighbors via links
- at each node there is a queue, called the *elastic buffer*
- received frames are added to the tail of the elastic buffer
- at each node, with each clock tick
  - a frame is removed from all elastic buffers at that node
  - a frame is sent on all outgoing links (hence frames are *conserved*)

Mechanism



- at each node the frequency of the oscillator is *controlled*
- if oscillator at node j is faster than that at i
  - j's elastic buffer will drain
  - i's elastic buffer will fill
- each node
  - observes the occupancy of its elastic buffers (one buffer per incoming link)
  - adjusts the oscillator frequency accordingly

Logical Synchrony



- marked frames from nodes 1 and 3 always arrive simultaneously at node 2
- an example of *logical synchrony*
- does not require clocks to be synchronized, but requires drift is not 'too large' to prevent elastic buffer overflow/underflow

Dynamic model

$$\dot{\theta}_i = \omega_i$$

$$\beta_{i \to j}(t) = \lfloor \theta_i(t - l_{i \to j}) \rfloor - \lfloor \theta_j(t) \rfloor + \lambda_{i \to j}$$

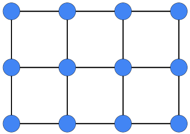
- $\lambda_{i \to j}$ is a constant, determined by clock offsets at boot
- buffer occupancy is (roughly) phase difference between clocks at each end of the link
- control objective: keep buffer occupancies close to steady-state, to prevent overflow/underflow

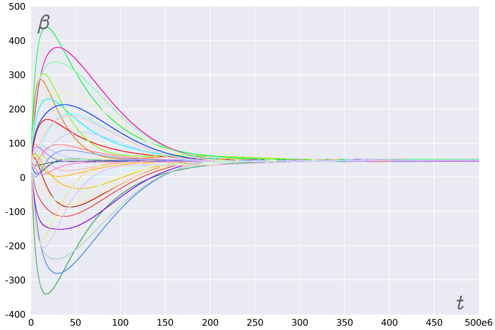
Controlling the frequency

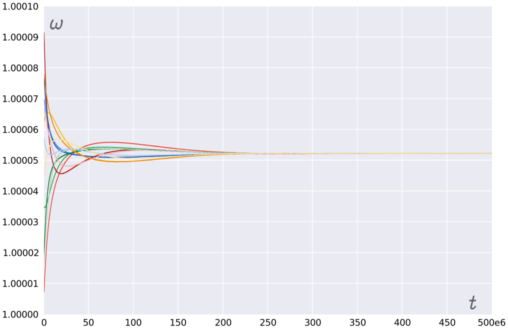
- at each node i, measure buffer occupancies $\beta_{j \to i}$ for all neighbours j (relative to a mid-point value β^{off})
- let r_i be the sum $r_i = \sum_{j \mid j \sim i} (\beta_{j \to i} - \beta^{\text{off}})$
- choose *frequency correction* using a control scheme
- for example, with proportional control $c_i = k_P r_i$ where k_P is the *proportional gain*
- set the frequency of the oscillator to be $\omega_i = c_i + \omega_i^{\text{u}}$
- ω_i^{u} is the *uncorrected frequency* of the oscillator, not known

Proportional-integral control

- ensures small steady state (offset removed) buffer occupancy







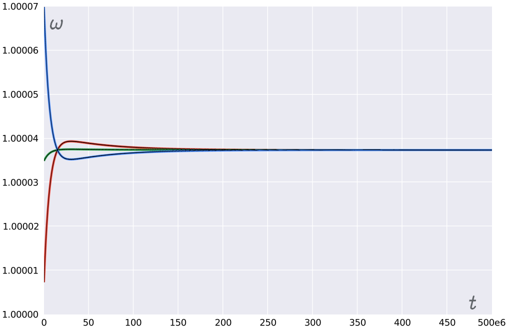
Fluid model

$$\dot{\bar{\theta}} = \omega$$

$$\bar{\beta} = -B^{\mathsf{T}}\bar{\theta}$$

$$r = B\bar{\beta}$$

$$\omega = c + \omega^{\mathrm{u}}$$



Closed-loop system

$$\begin{bmatrix} \dot{\bar{\theta}} \\ \dot{\xi} \end{bmatrix} = \begin{bmatrix} -k_P B B^\top & k_I I \\ -B B^\top & 0 \end{bmatrix} \begin{bmatrix} \bar{\theta} \\ \xi \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} e$$

$$\bar{\beta} = -B^\top \bar{\theta}$$

$$r = -B B^\top \bar{\theta}$$

$$\omega = k_I \xi - k_P B B^\top \bar{\theta} + e$$

- 2-dimensional uncontrollable/unobservable subspace (all nodes equal frequencies)
- remaining dynamics are stable

Performance

- we are interested in keeping certain quantities small
  - difference between frequency and steady-state frequency
  - difference between buffer occupancy and initial buffer occupancy
- we will measure these using the L_2 norm

$$\|\omega\| = \left(\int_0^\infty \sum_{i=1}^{n} \omega_i(t)^2 \, dt \right)^{\frac{1}{2}}$$

Results

$$||\omega - \omega^{\mathrm{ss}}||^2 = \frac{1}{2k_P} e^\top L^\dagger e$$
$$||\delta||^2 = \frac{1}{2k_P k_I} e^\top L^\dagger e$$

- here L is the graph Laplacian $L = BB^\top$
- separates effect of graph from effect of controller parameters
- adding edges or increasing gain always improves performance

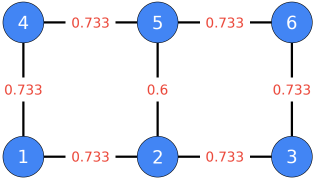
Resistance interpretation

- construct a circuit from the graph, with 1Ω resistors along each edge
- let R_{ij} be the resistance between nodes i and j, then

$$R_{ij} = (\mathbb{e}_i - \mathbb{e}_j)^\top L^\dagger (\mathbb{e}_i - \mathbb{e}_j)$$

- *Rayleigh monotonicity:* adding an edge cannot increase any R_{ij}

Example: Resistance



- $R_{15} \approx 0.933,\ R_{16} = 1.34,\ R_{13} \approx 1.333$

Two disequilibriated frequencies

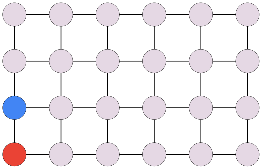
- Suppose frequency at node a is $1+\alpha$, and at node b is $1-\alpha$, with all other nodes at frequency 1
- then $e^\top L^\dagger e = 2\alpha R_{ab}$ and so

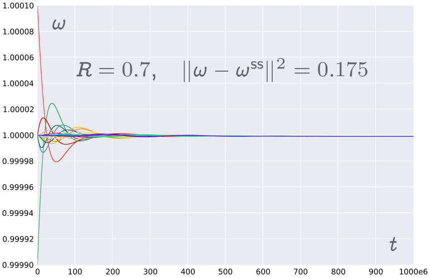
$$||\omega - \omega^{\text{ss}}||^2 = \frac{1}{2k_P} e^\top L^\dagger e = \frac{\alpha R_{ab}}{k_P}$$

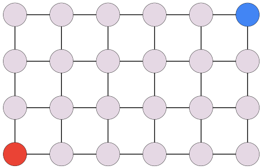
$$||\delta||^2 = \frac{1}{2k_P k_I} e^\top L^\dagger e = \frac{\alpha R_{ab}}{k_P k_I}$$

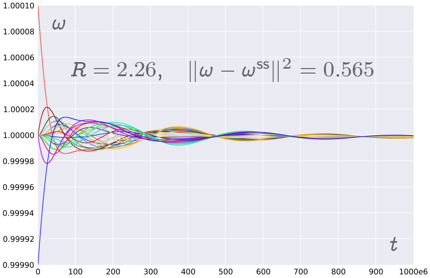
Two disequilibriated frequencies

Here $k_P = 2 \times 10^{-8}$ and $\alpha = 10^{-4}$

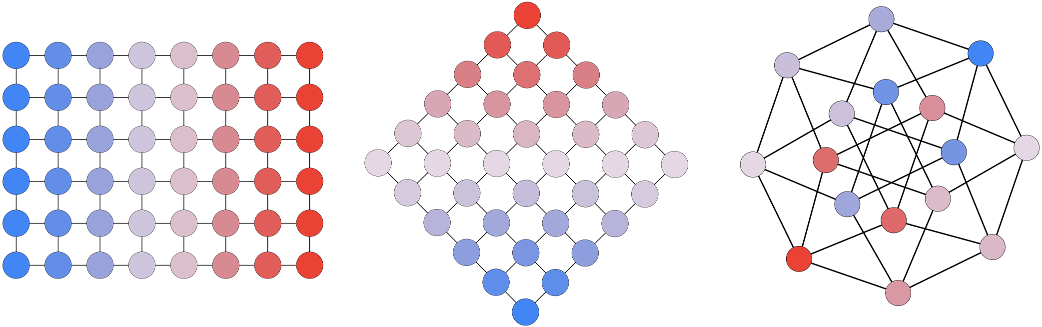








Worst-case error distributions



- we know $||\omega - \omega^{\text{ss}}||^2 = \dfrac{1}{2k_P} e^\top L^\dagger e$
- we can find the worst-case error distribution, by maximizing $e^\top L^\dagger e$ over all e with $||e|| \leq 1$
- worst-case e is the eigenvector of L corresponding to the second-smallest eigenvalue

Summary

- bittide is a system for producing logical synchrony
- underlying control system regulates node frequencies
- certain performance measures capture via resistance distance of graph