Predictable Self-Organization with Computational Fields

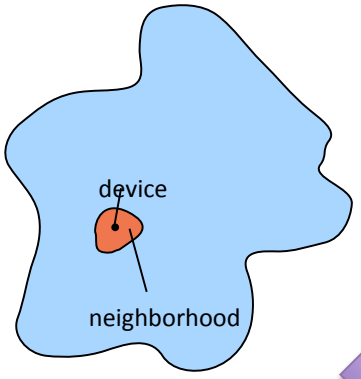
Jacob Beal, Mirko Viroli

Tutorial at 8th IEEE SASO
Imperial College London
September 2014

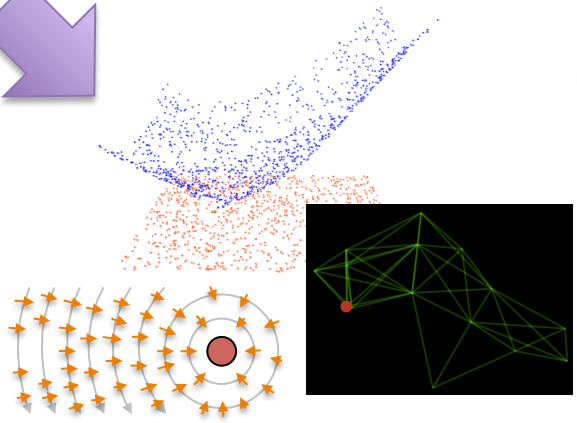
Raytheon
BBN Technologies

Aggregate Programming: The Big Picture

Raytheon BBN Technologies



Field-Based Computing Models



Self-Organization Building Blocks

Lots of distributed applications!

Our Promises to You

Raytheon
BBN Technologies

By the end of this tutorial you will…

- know about aggregate programming, field calculus, and continuous space-time abstractions
- be able to write programs in Proto
- understand how field computation can support predictable engineering of complex adaptive systems
- understand how to build complex self-organizing systems by composing building-block algorithms

Schedule:

Raytheon
BBN Technologies

14:00 – 14:30	Aggregate Programming & Spatial Computing	*Jake*
14:30 – 15:30	Field Calculus	*Mirko*
	Coffee Break	
16:00 – 16:20	Ensuring Self-stabilisation	*Mirko*
16:20 – 17:20	Building Blocks & Consistency	*Jake*
17:20 – 17:30	Summary and Open Problems	*Jake*

Aggregate Programming and Spatial Computing

Jacob Beal, Mirko Viroli

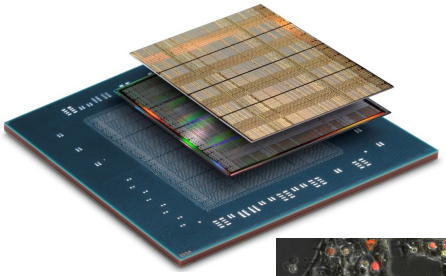
Tutorial at 8th IEEE SASO
Imperial College London
September 2014

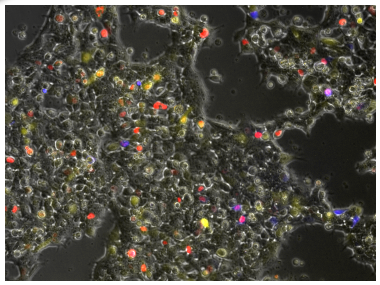
Raytheon
BBN Technologies

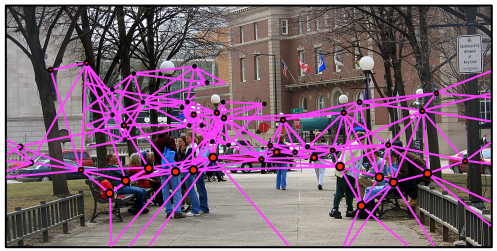
Don’t micromanage your networks!

Raytheon
BBN Technologies

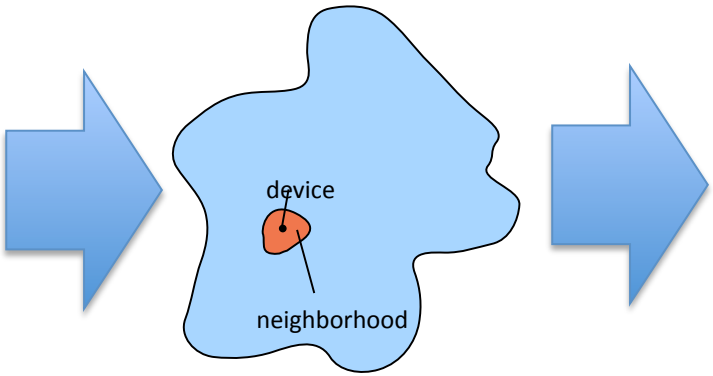
Emerging Computational Platforms







Computational Field Programming Models

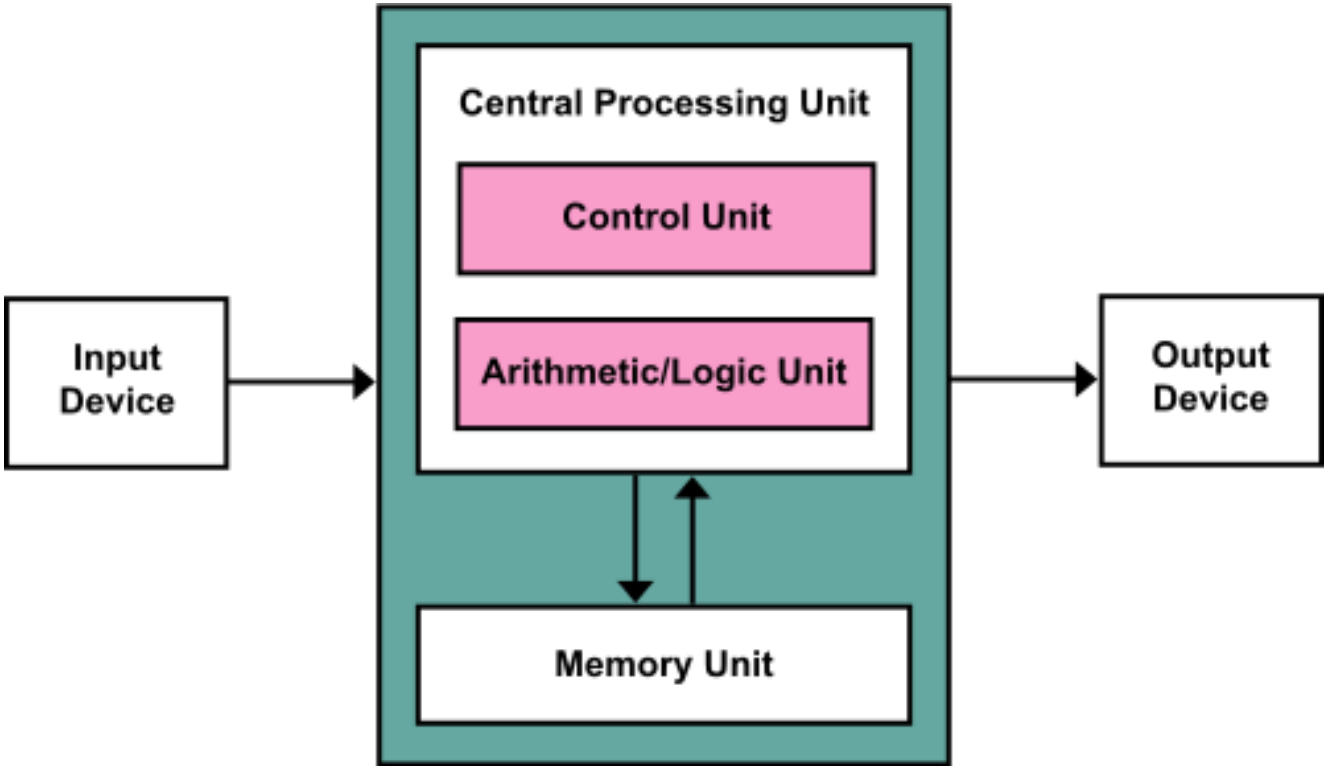


Inherently Resilient Distributed Systems

Pay a little efficiency, get a lot of programmability and resilience

Traditional Monolithic Computing

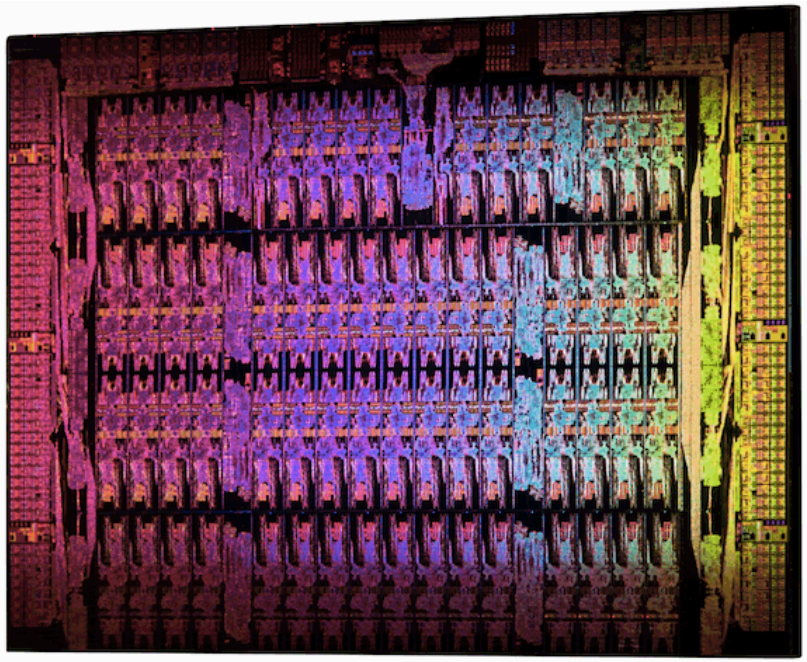
Raytheon
BBN Technologies



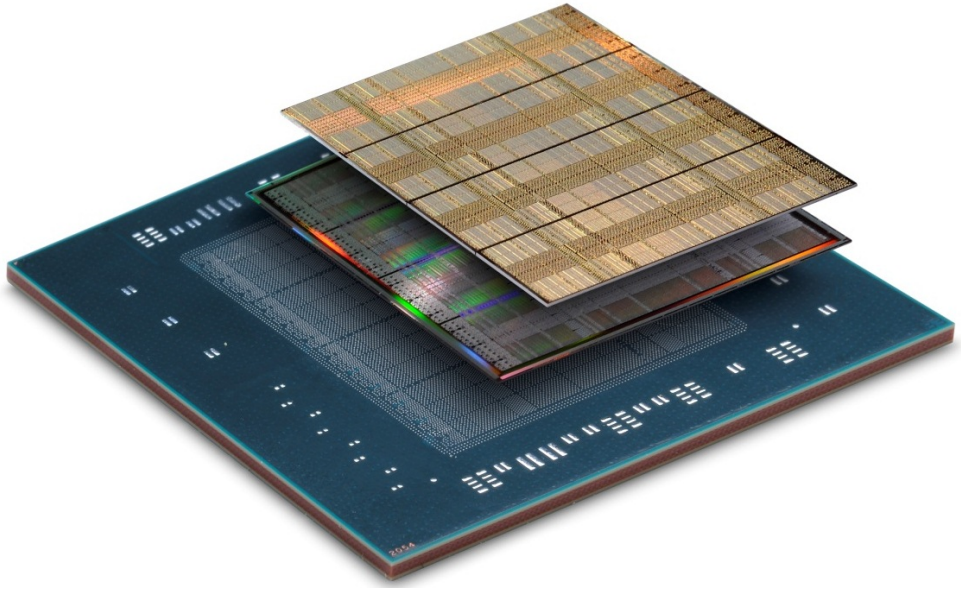
The venerable von Neumann model is breaking down in several ways…

The End of Moore's Law

Raytheon
BBN Technologies



Intel Xeon Phi: 61 cores

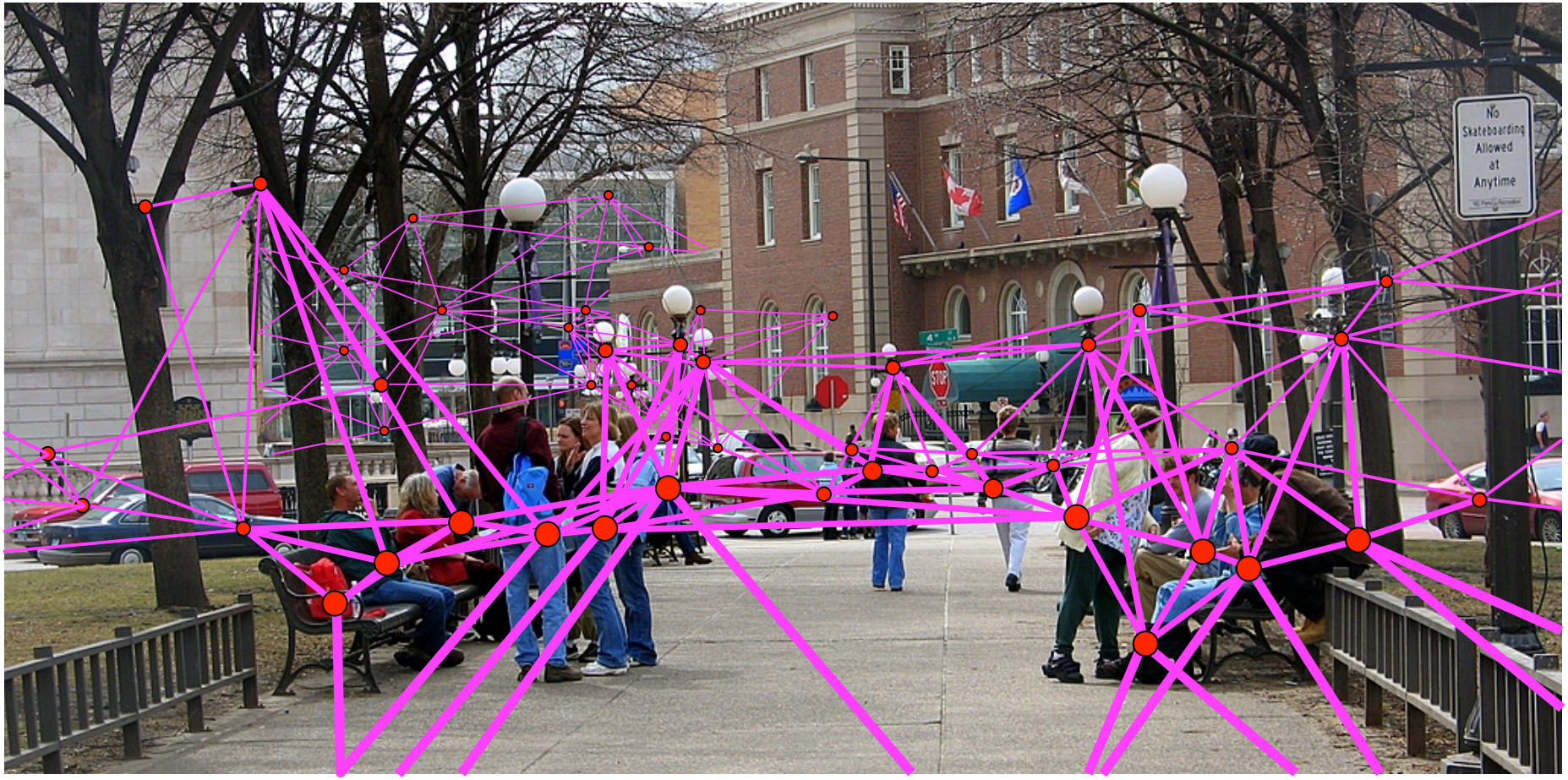


Xilinx Virtex-7: 2M Logic cells

High-performance computing = mesh

Everything is a wireless computer

Raytheon
BBN Technologies

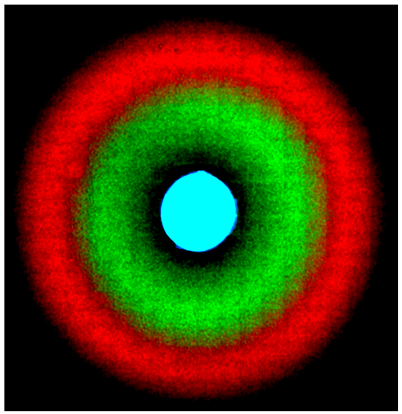


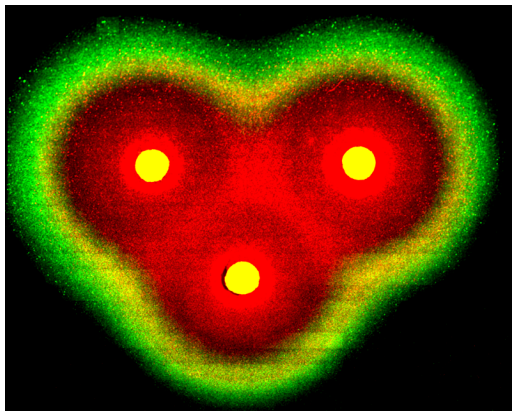
New Computational Materials

Raytheon
BBN Technologies

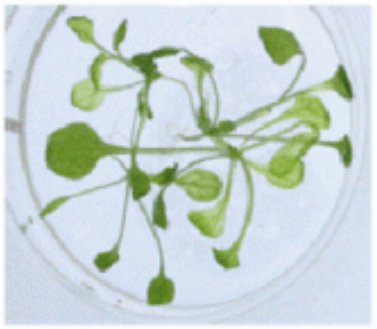
- Synthetic Biology:

[Levskaya]

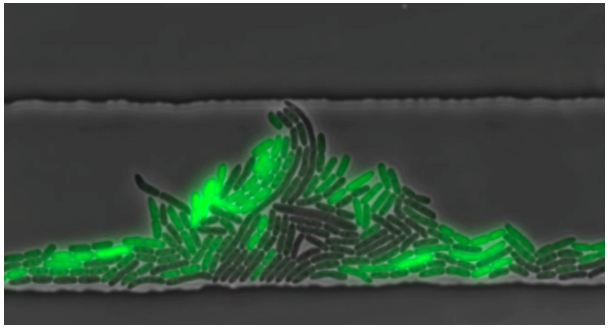




[Weiss]



[Medford]

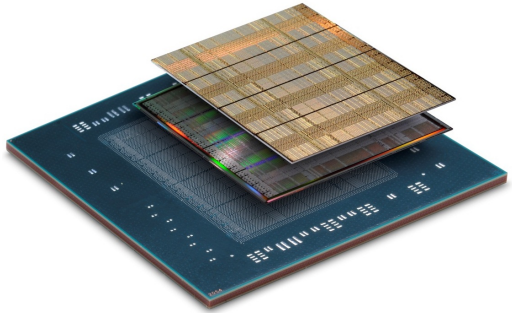


[Hasty]

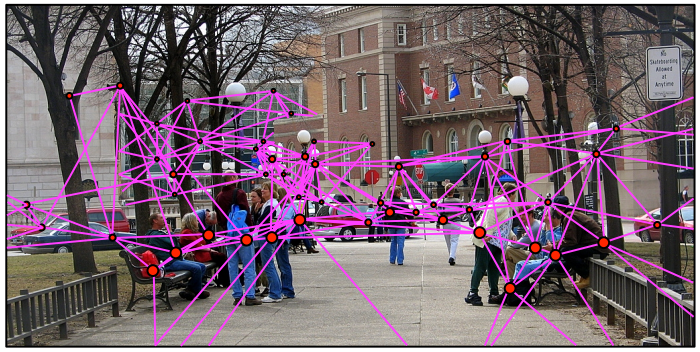
Other emerging areas too, including nanoassembly, active materials…

Fundamentally different models

Raytheon
BBN Technologies



Isolate Systems
Extremely High FLOPs



High Dispersion
Moderate FLOPs

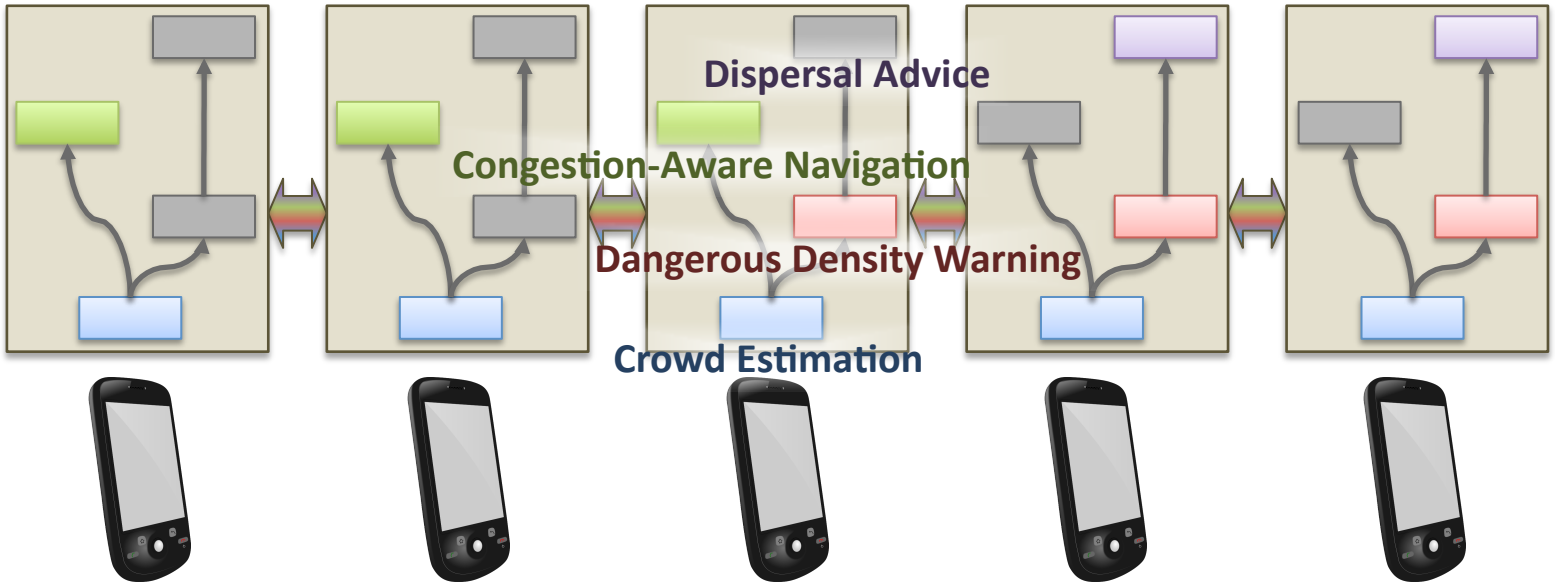
High Resolution Sense/Act
Abysmal FLOPs

How can we program aggregates adaptively & efficiently?

Are there commonalities that cross substrates?

Device-Centric Programming

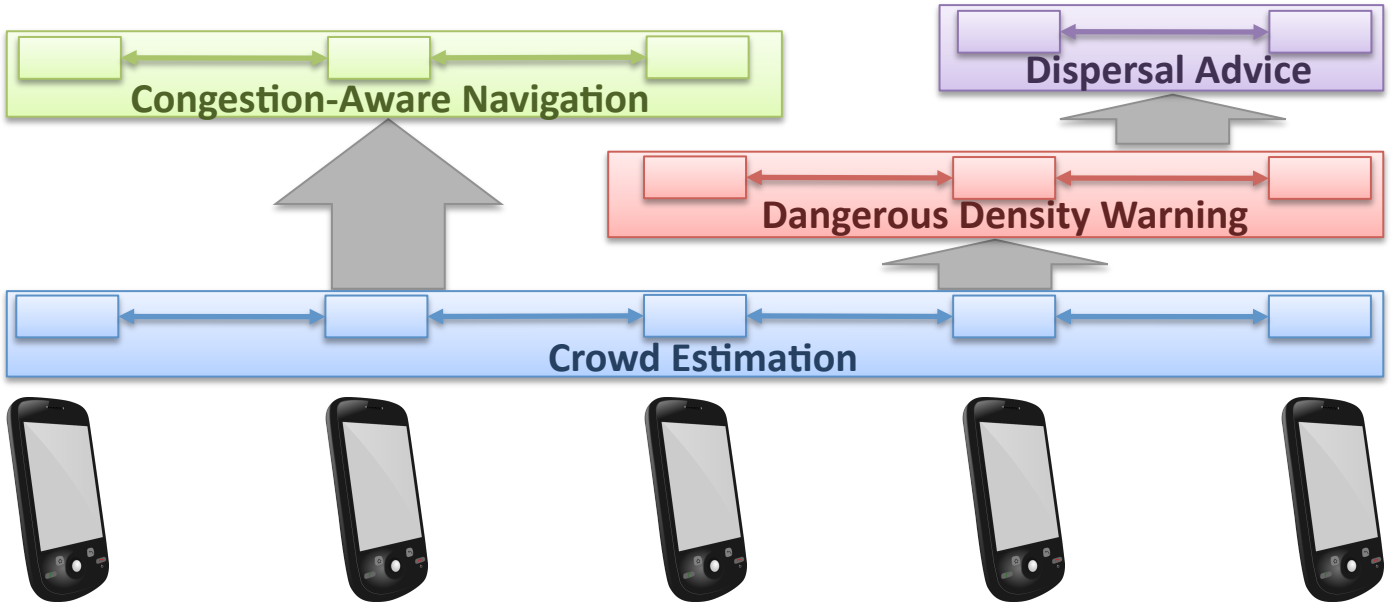
Raytheon
BBN Technologies



- Explicit design of adaptation and communication
- Complex per-device multi-service application
- Intractable to ensure correct behavior

Aggregate Programming

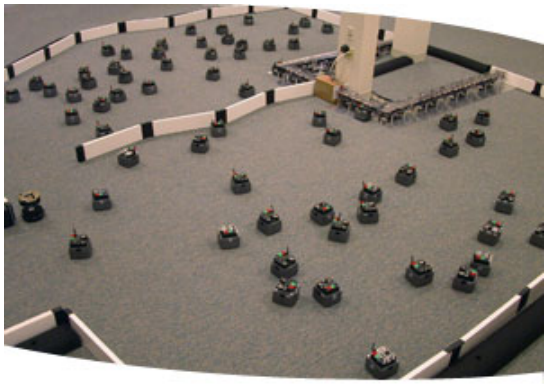
Raytheon
BBN Technologies



- Implicit adaptation and communication
- Code each collective service independently
- Compose via scope and information flow

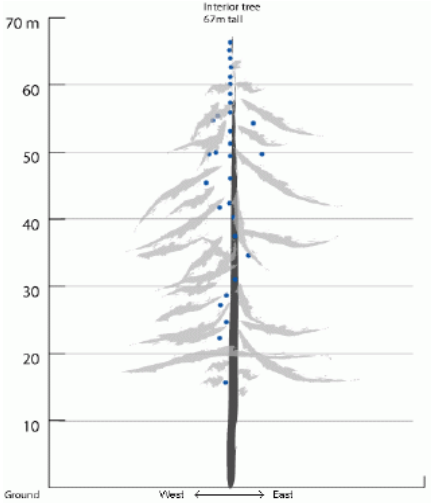
Spatial Computers

Raytheon
BBN Technologies

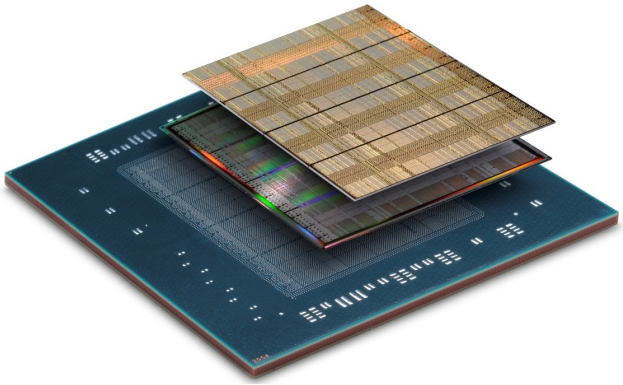


Robot Swarms

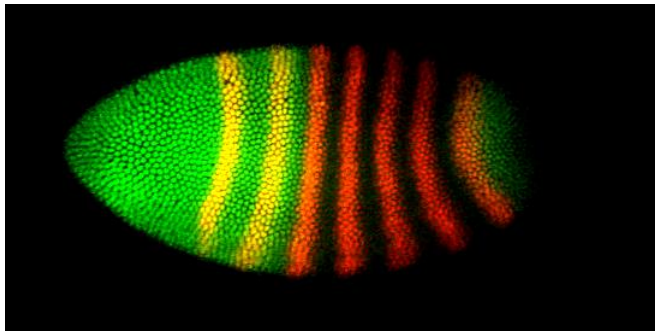
Biological Computing



Sensor Networks



Reconfigurable Computing



Cells during Morphogenesis

Modular Robotics

More formally...

Raytheon
BBN Technologies

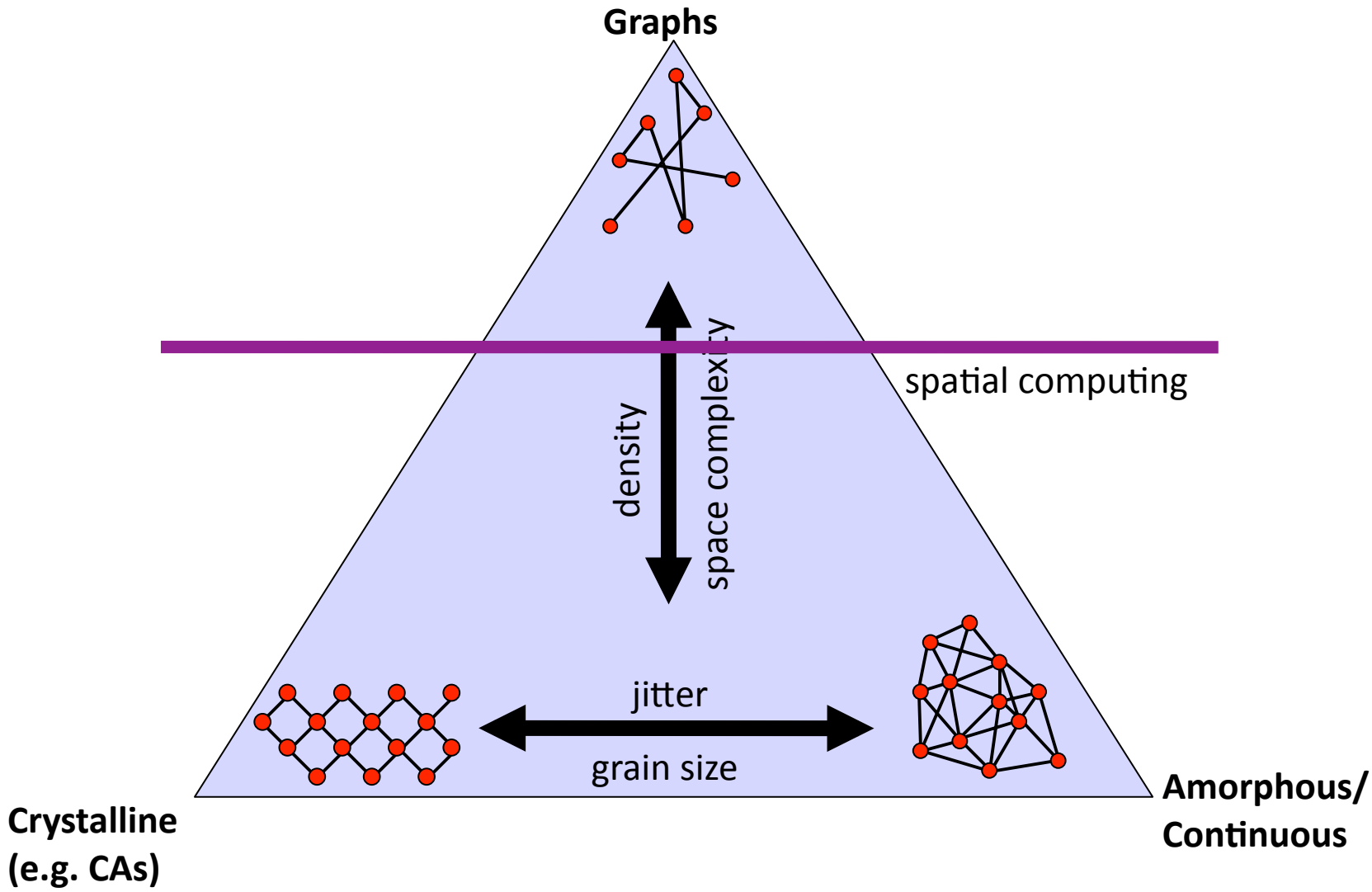
- A spatial computer is a collection of computational devices distributed through a physical space in which:
  - the difficulty of moving information between any two devices is strongly dependent on the distance between them, and
  - the “functional goals” of the system are generally defined in terms of the system's spatial structure

More formally...

Raytheon
BBN Technologies

- A spatial computer is a collection of computational devices distributed through a physical space in which:
  - the difficulty of moving information between any two devices is strongly dependent on the distance between them, and
  - the “functional goals” of the system are generally defined in terms of the system's spatial structure

Notice the ambiguities in the definition


Graphs
spatial computing
density
space complexity
jitter
grain size
Crystalline
(e.g. CAs)
Amorphous/
Continuous

(w. Dan Yamins)

Example: Services for Mass Events

Raytheon
BBN Technologies

Example: Managing Crowd Danger

Raytheon
BBN Technologies

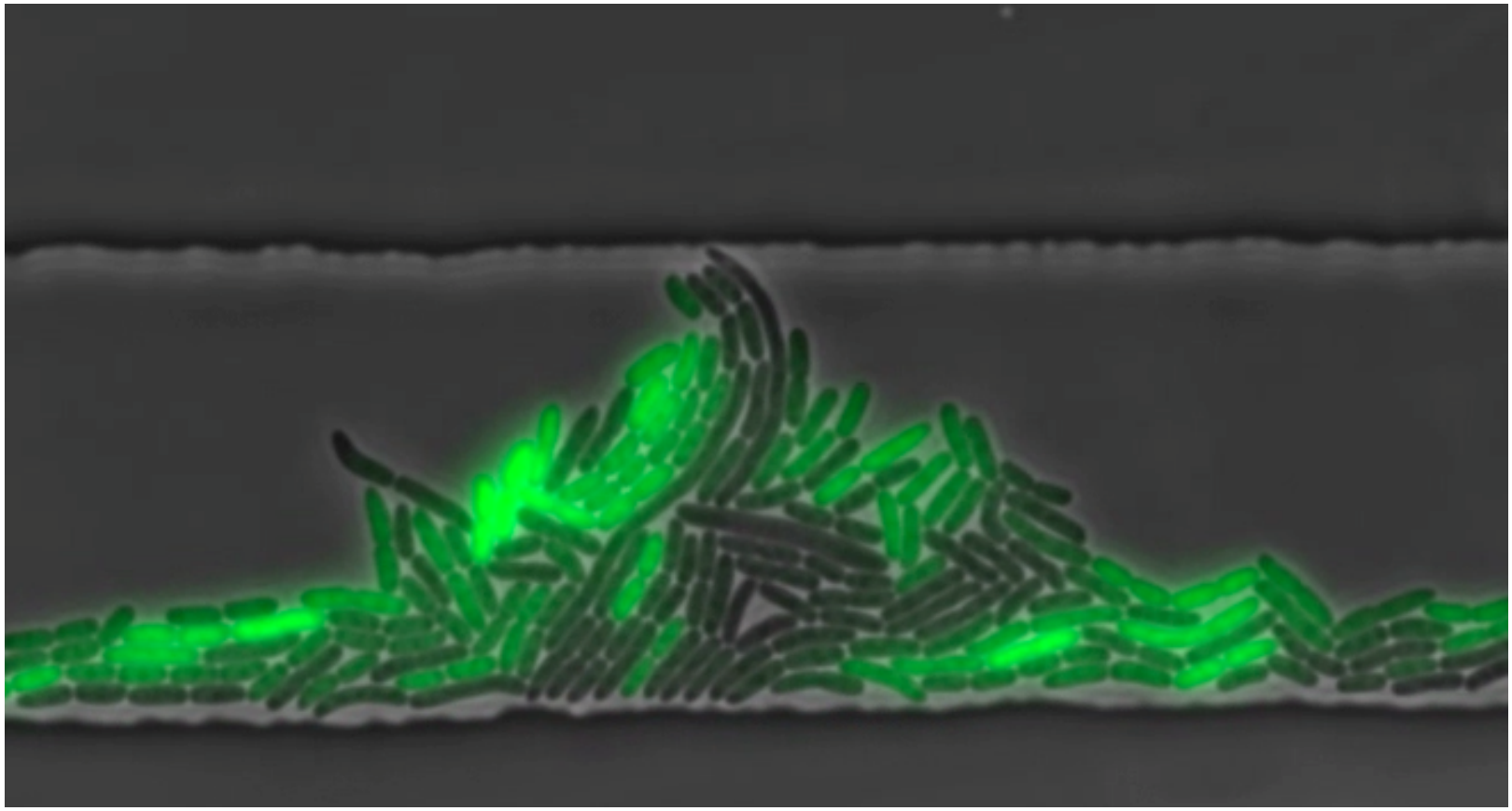
Example: Agent-Based Simulations

Raytheon
BBN Technologies

[Beal et al., 2012]

Example: Cellular Oscillator

Raytheon
BBN Technologies



[Danino et al., 2010]

How can we program these?

Raytheon
BBN Technologies

- Desiderata for approaches:
  - Simple, easy to understand code
  - Robust to errors, adapt to changing environment
  - Scalable to potentially vast numbers of devices
  - Take advantage of spatial nature of problems

Myriad proposed approaches

Raytheon
BBN Technologies

A small sample…

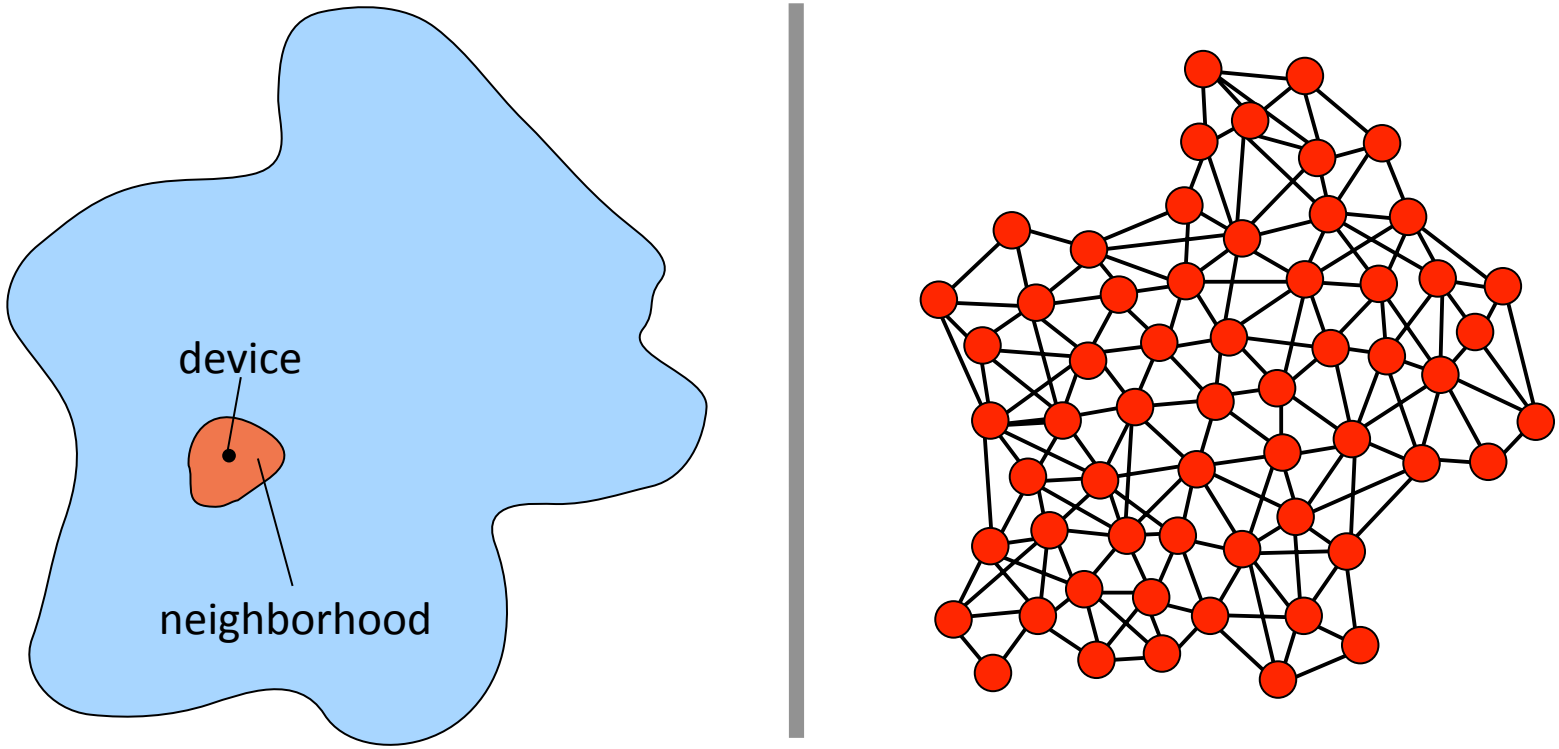
Many Domains: Sensor Networks, Agent systems, Biological Modeling, Synthetic Biology, Pervasive Computing, Amorphous, Swarm/Modular Robotics, High-Peformance Computing, Reconfigurable Computing, …

How do we understand and synthesize them all?

[Beal et al., ’12]

Space/Network Type #1: Sampling

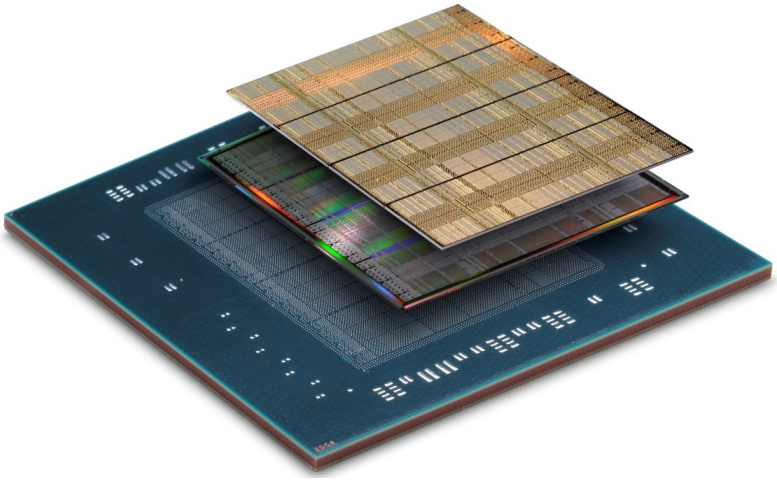
Raytheon
BBN Technologies



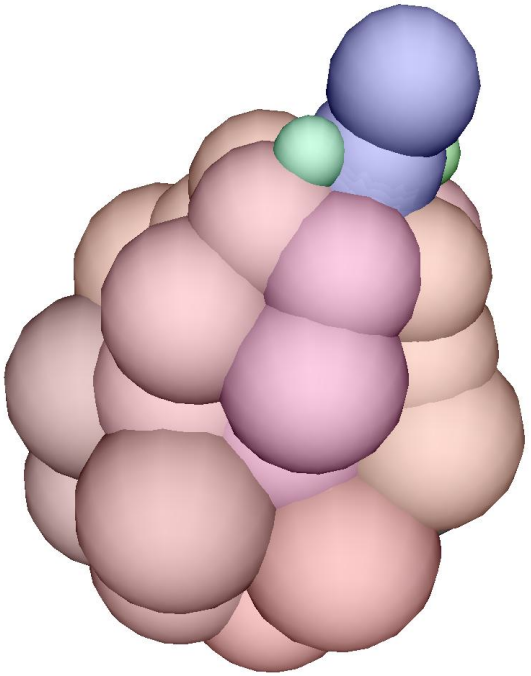
e.g. sensor nets, pervasive systems, swarm robots

Space/Network Type #2: Cellular

Raytheon
BBN Technologies



Crystalline

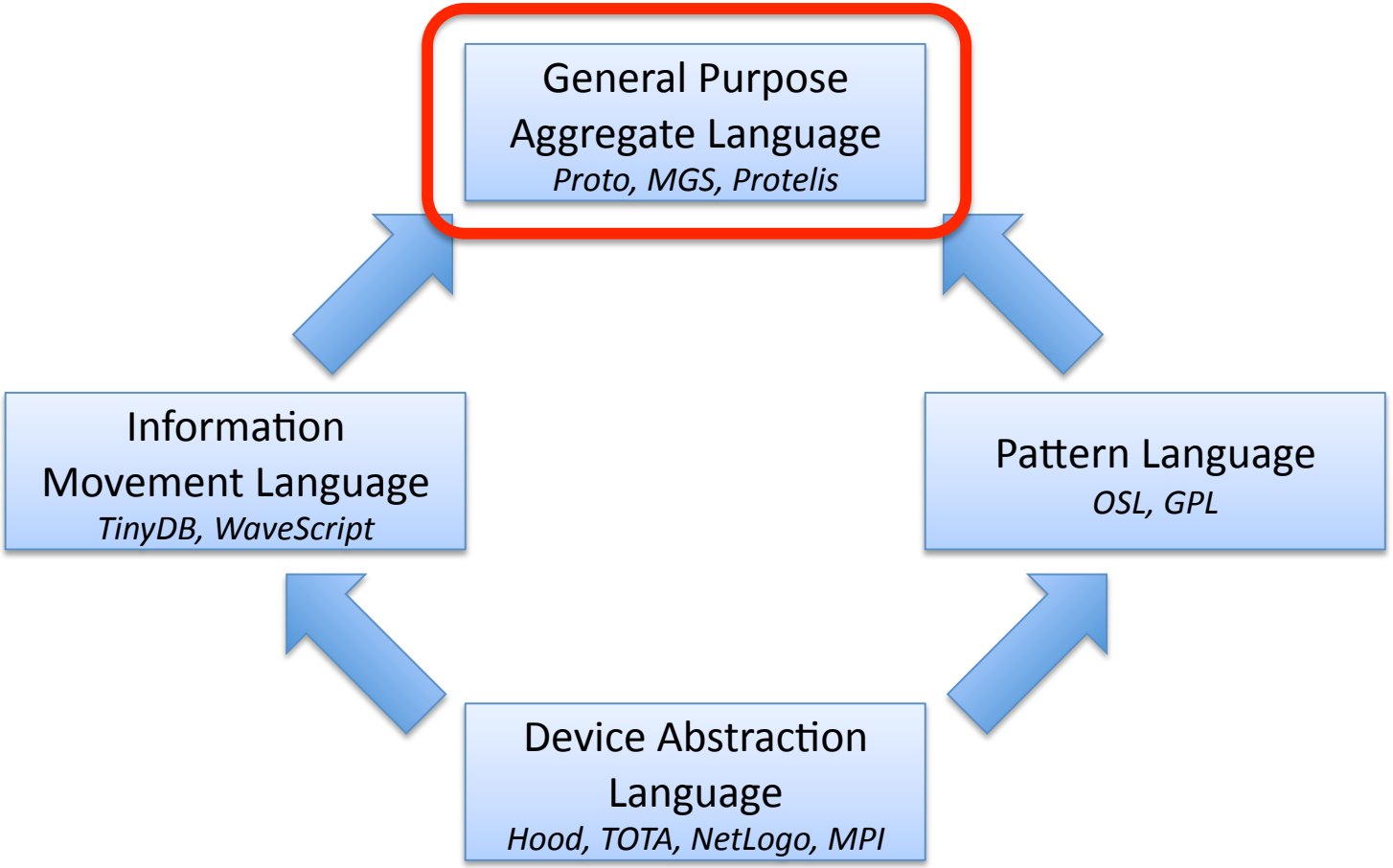


Amorphous

e.g. processing fabrics, biologicals, modular robots

Four General Language Classes

Raytheon
BBN Technologies

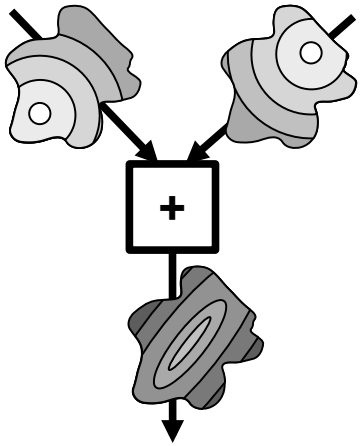


[Beal et al., '12]

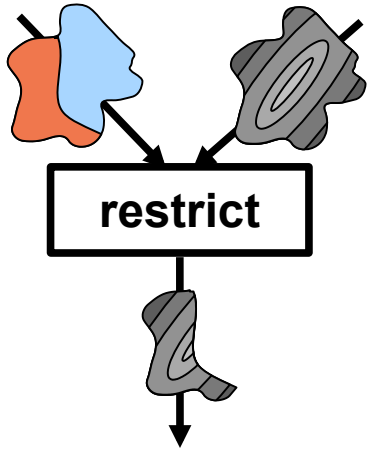
Field Calculus:

Raytheon
BBN Technologies

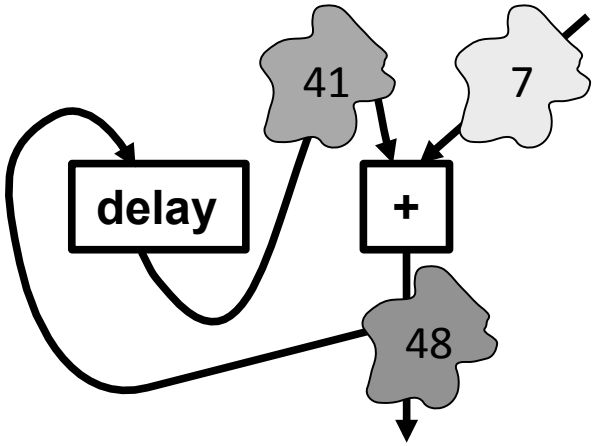
Pointwise



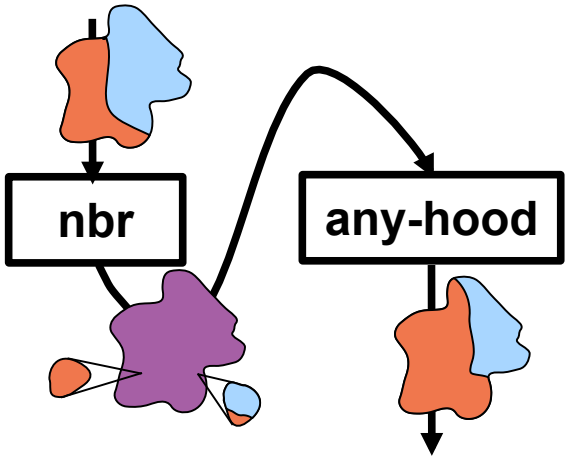
Restriction: `if`



Feedback: `rep`



Neighborhood: `nbr`

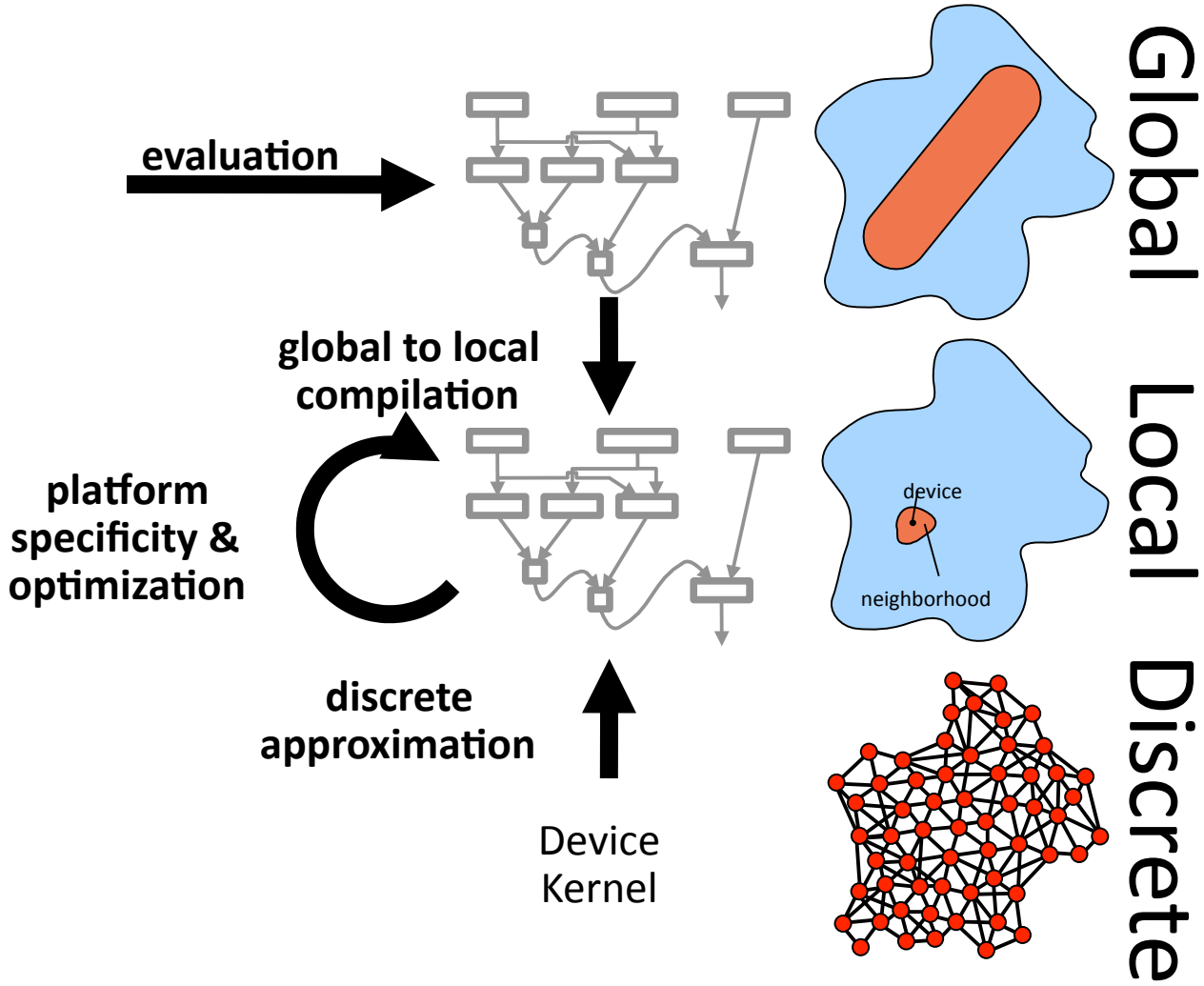


[Viroli et al., ’13]

Implementation: Proto

Raytheon
BBN Technologies

```
(def gradient (src) ...)
(def distance (src dst) ...)
(def dilate (src n)
  (<= (gradient src) n))
(def channel (src dst width)
  (let* ((d (distance src dst))
         (trail (<= (+ (gradient src)
                       (gradient dst))
                    d)))
    (dilate trail width)))
```



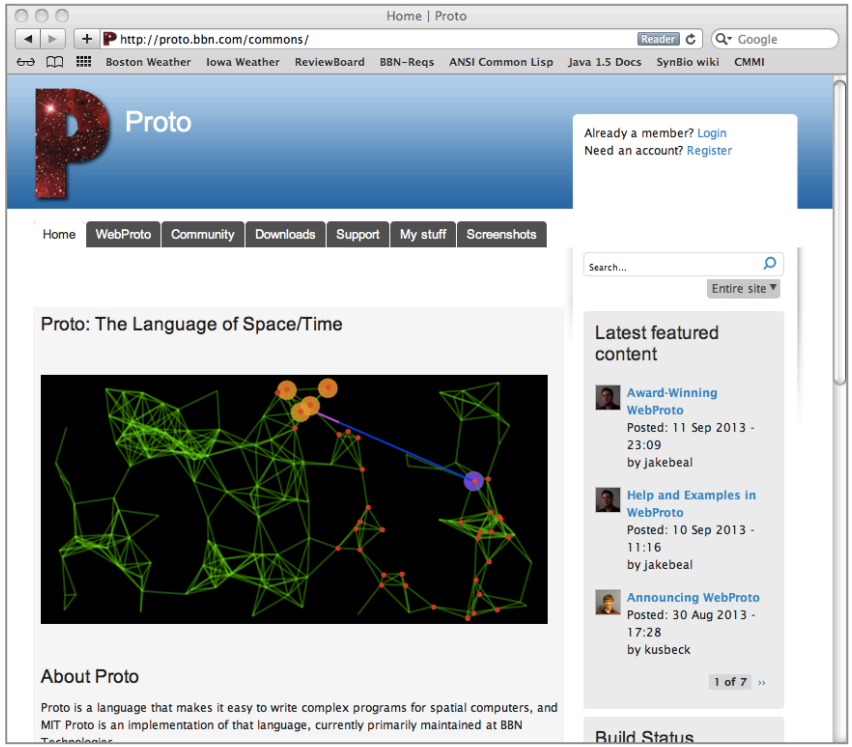
[Beal & Bachrach, '06]

Proto available online!

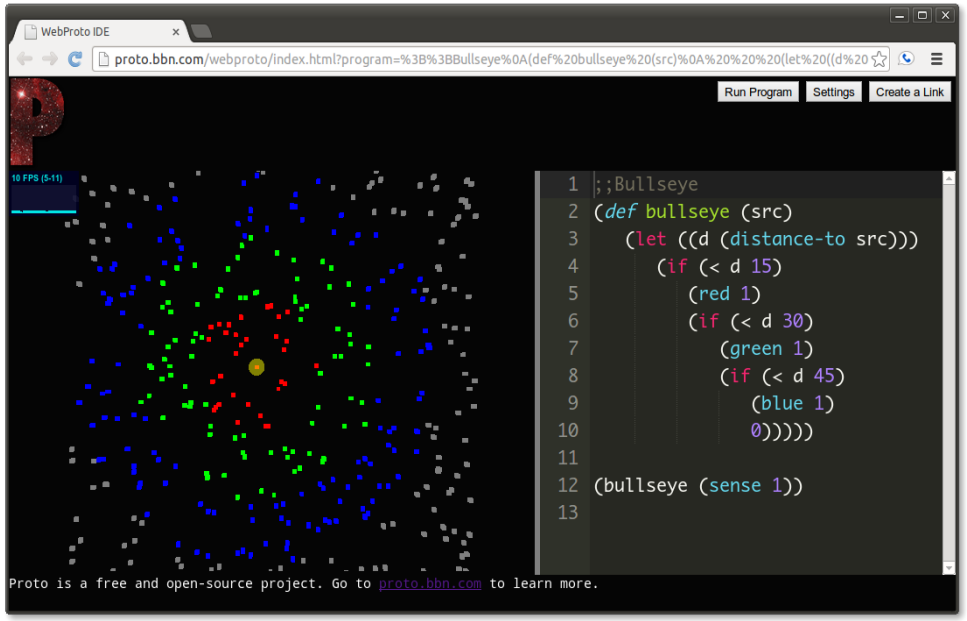
Raytheon BBN Technologies

http://proto.bbn.com/

Free & Open Software



WebProto



From Principle to Practice:

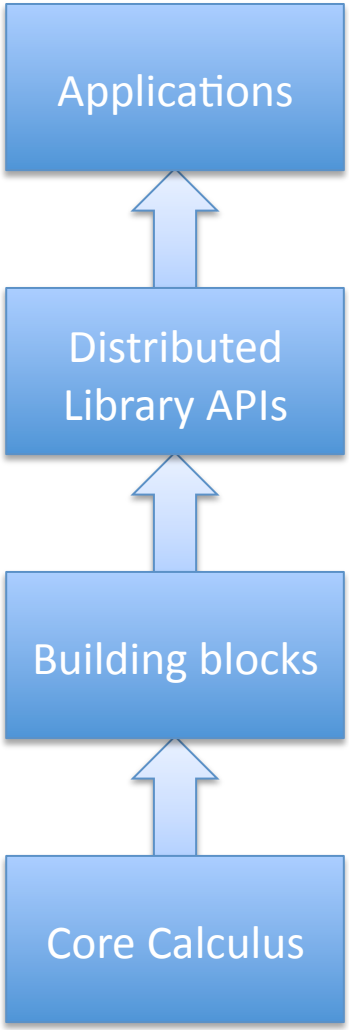
Raytheon
BBN Technologies

Transparent use of robust distributed algorithms

Programmer level: lots of useful, intuitive methods Domain-specific APIs

Provable robustness, scalability, composability

Provable universality, aggregate/local relation



unlimited uses

many algorithms

5-20 combinators

5 operators

Summary

Raytheon
BBN Technologies

- Major technological trends are all driving towards a world filled with distributed systems
- Aggregate programming aims at rapid and reliable engineering of complex distributed systems
- Field calculus provides a solid mathematical foundation for predicting aggregate behaviors
- Aggregates are often closely tied to space, and geometry is a good source of "building blocks"
- Methods are useful for many types of application

Bibliography

Raytheon
BBN Technologies

- Beal J, Dulman S, Usbeck K, Viroli M, Correll N. Organizing the Aggregate: Languages for Spatial Computing. In: Mernik M, editor. Formal and Practical Aspects of Domain-Specific Languages: Recent Developments. IGI Global; 2013. p. 436–501.
- Viroli M, Damiani F, Beal J. A Calculus of Computational Fields. In: Canal C, Villari M, editors. Advances in Service-Oriented and Cloud Computing. vol. 393 of Communications in Computer and Information Sci. Springer Berlin Heidelberg; 2013. p. 114–128.
- Beal J, Bachrach J. Infrastructure for Engineered Emergence in Sensor/Actuator Networks. IEEE Intelligent Systems. 2006 March/April;21:10–19.