A Theory Approach to Local-to-Global Algorithms in Spatial Multi-Agent Systems

CS266, Fall 2007
Dan Yamins

Session II: 12.06.2007
Recall the definitions of the model from last time:
Recall the definitions of the model from last time:

$G$ Underlying Geometry: a bunch of graphs.
Recall the definitions of the model from last time:

\( \mathcal{G} \) Underlying Geometry: a bunch of graphs.

\( S \) State set of size \( m \). (colors)
Recall the definitions of the model from last time:

\( G \) Underlying Geometry: a bunch of graphs.

\( S \) State set of size m. (colors)

\( X \) Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set \( S \).
Recall the definitions of the model from last time:

$G$ Underlying Geometry: a bunch of graphs.

$S$ State set of size $m$. (colors)

$X$ Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set $S$.

$R$ Communication radius
Recall the definitions of the model from last time:

\( G \) Underlying Geometry: a bunch of graphs.

\( S \) State set of size \( m \). (colors)

\( X \) Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set \( S \).

\( R \) Communication radius

\( F \) Local dynamical update rules. Input: \( R \)-neighborhoods
Recall the definitions of the model from last time:

- **G** Underlying Geometry: a bunch of graphs.
- **S** State set of size m. (colors)
- **X** Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set S.
- **R** Communication radius
- **F** Local dynamical update rules. Input: R-neighborhoods  
  Output: New state in S.
Recall the definitions of the model from last time:

$G$ Underlying Geometry: a bunch of graphs.

$S$ State set of size m. (colors)

$X$ Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set $S$.

$R$ Communication radius

$F$ Local dynamical update rules. Input: R-neighborhoods
Output: New state in S.

$S$ Timing model: agents called in various orders iteratively apply the local rule, generating trajectories.
Recall the definitions of the model from last time:

\( G \) Underlying Geometry: a bunch of graphs.

\( S \) State set of size \( m \). (colors)

\( X \) Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set \( S \).

\( R \) Communication radius

\( F \) Local dynamical update rules. Input: R-neighborhoods Output: New state in \( S \).

\( T \) Timing model: agents called in various orders iteratively apply the local rule, generating trajectories.

\( T \) Pattern
Recall the definitions of the model from last time:

**G** Underlying Geometry: a bunch of graphs.

**S** State set of size \( m \). (colors)

**X** Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set \( S \).

**R** Communication radius

**F** Local dynamical update rules. Input: \( R \)-neighborhoods

Output: New state in \( S \).

**S** Timing model: agents called in various orders iteratively apply the local rule, generating trajectories.

**T** Pattern

\[ \implies F \] Robust Solutions: rules whose trajectories always converge to \( T \) from all initial configurations and under all call orders.
Recall the definitions of the model from last time:

$G$ Underlying Geometry: a bunch of graphs.

$S$ State set of size $m$. (colors)

$X$ Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set $S$.

$R$ Communication radius

$F$ Local dynamical update rules. Input: $R$-neighborhoods


$T$ Timing model: agents called in various orders iteratively apply the local rule, generating trajectories.

Robust Solutions: rules whose trajectories always converge to $T$ from all initial configurations and under all call orders.
Recall the definitions of the model from last time:


$S$: State set of size m. (colors)

$X$: Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set $S$.

$R$: Communication radius

$F$: Local dynamical update rules. Input: R-neighborhoods Output: New state in S.

$S$: Timing model: agents called in various orders iteratively apply the local rule, generating trajectories.

$T$: Pattern

$\Rightarrow F$: Robust Solutions: rules whose trajectories always converge to $T$ from all initial configurations and under all call orders.
Recall the definitions of the model from last time:

**Statics**

- **G** Underlying Geometry: a bunch of graphs.
- **S** State set of size m. (colors)
- **X** Configurations. Agents were nodes in the graphs; graph gives spatial relations. Internal states were node-labels from the set S.

**Dynamics**

- **R** Communication radius
- **F** Local dynamical update rules. Input: R-neighborhoods
  
  
  Output: New state in S.

**Task (or “functionality”)**

- **T** Pattern

\[ \Rightarrow F \]

Robust Solutions: rules whose trajectories always converge to T from all initial configurations and under all call orders.
Last time ....

... the model.

Today ...

... some results.
Local Checkability
Local Checkability

Let’s take the simple 1-D repeat pattern $T_{10}$:
Local Checkability

Let’s take the simple 1-D repeat pattern $T_{10}$:

![Diagram of a 1-D repeat pattern with 1s and 0s]

Problem: Find a nearest-neighbor solution to this pattern
Let’s take the simple 1-D repeat pattern $T_{10}$:

Problem: Find a nearest-neighbor solution to this pattern

Answer:

$$F(B) = \begin{cases} 
1 - B(-1), & B \neq \text{left-end agent} \\
1, & B = \text{left-end agent}
\end{cases}$$
Local Checkability

Now consider the repeat pattern $T_{1000}$:
Local Checkability

Now consider the repeat pattern $T_{1000}$:

Can this pattern be solved robustly with a nearest-neighbor rule?
Local Checkability

Now consider the repeat pattern $T_{1000}$:

Can this pattern be solved robustly with a nearest-neighbor rule?

Answer: No. Because the with a radius 1 rule, 000 would have to be a fixed state.
Local Checkability
Local Checkability

Now take the proportionate pattern:
Local Checkability

Now take the proportionate pattern:

Problem: What is the smallest radius that will solve T?
Local Checkability

Now take the proportionate pattern:

Problem: What is the smallest radius that will solve T?

Answer: Infinity. There is no solution.
Local Checkability

Now take the proportionate pattern:

Problem: What is the smallest radius that will solve $T$?

Answer: Infinity. There is no solution.

Because this configuration:
Local Checkability

Now take the proportionate pattern:

0 → 0 → 0 → 0 → 1 → 1 → 1 → 1

Problem: What is the smallest radius that will solve T?

Answer: Infinity. There is no solution.

Because this configuration:

0 → 0 → 0 → 0 → 0 → 1 → 1 → 1 → 1 → 1 → 1 → 1 → ··· → 1
Local Checkability

Now take the proportionate pattern:

0 → 0 → 0 → 0 → 1 → 1 → 1 → 1

Problem: What is the smallest radius that will solve T?

Answer: Infinity. There is no solution.

Because this configuration:

0 → · · · → 0 → 0 → 0 → 0 → 1 → 1 → 1 → 1 → · · · → 1

Will be indistinguishable from this one:

⇒ r(F)
Local Checkability

Now take the proportionate pattern:

```
  0 → 0 → 0 → 0 → 1 → 1 → 1 → 1
```

Problem: What is the smallest radius that will solve T?

Answer: Infinity. There is no solution.

Because this configuration:

```
  0 → 0 → 0 → 0 → 1 → 1 → 1 → 1
```

Will be indistinguishable from this one:

```
  0 → 0 → 0 → 0 → 0 → 0 → 0 → 1 → 1
```

\[ r(F) \]
Local Checkability
Local Checkability

Definition. A function $\Theta : B_r \rightarrow \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)} (\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.
Local Checkability

**Definition.** A function $\Theta : B_r \to \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)}(\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.
Definition. A function $\Theta : B_r \rightarrow \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \land_{i \in V(X)} (\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.
Local Checkability

**Definition.** A function $\Theta : \mathcal{B}_r \rightarrow \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)} (\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and

- $T \cap \mathcal{C}_n \neq \emptyset \Rightarrow$ there is $X \in \mathcal{C}_n$ such that $\Theta[X]$ holds.
**Local Checkability**

**Definition.** A function $\Theta : B_r \to \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)}(\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and

- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.

\[ \begin{array}{cccccccc}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\end{array} \ldots \]
Local Checkability

**Definition.** A function $\Theta : B_r \to \{0,1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)} (\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.

**Example.** The pattern $T_{1000}$ has a radius-2 local check scheme.
Local Checkability

Definition. A function $\Theta : B_r \rightarrow \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)}(\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.

Example. The pattern $T_{1000}$ has a radius-2 local check scheme.

Let $T_\Theta$ be the pattern generated by $\Theta$. 
Local Checkability

Definition. A function $\Theta : B_r \rightarrow \{0, 1\}$ is a local check scheme for pattern $T$ if

- $\Theta[X] = \bigwedge_{i \in V(X)} (\Theta(B_r(i, X)) = 1) \Rightarrow X \in T$ and
- $T \cap C_n \neq \emptyset \Rightarrow$ there is $X \in C_n$ such that $\Theta[X]$ holds.

Example. The pattern $T_{1000}$ has a radius-2 local check scheme.

Let $T_\Theta$ be the pattern generated by $\Theta$.

Let LCR(T) denote the minimal radius of a check scheme for it -- this is $T$’s “local check radius.” $T$ is “locally checkable” if LCR(T) is finite.
Local Checkability
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution:
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution:

**Proposition.** *If $F$ is a robust solution to 1-dimensional pattern $T$, then*

\[ r(F) \geq LCR(T). \]
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** If $F$ is a robust solution to 1-dimensional any pattern $T$, then $r(F) \geq LCR(T)$.

**Proposition.** If $F$ is a robust solution to 1-dimensional pattern $T$, then $r(F) \geq LCR(T)$. 
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** *If* $F$ *is a robust solution to 1-dimensional any pattern* $T$, *then* $r(F) > LCR(T)$.

**Proposition.** *If* $F$ *is a robust solution to 1-dimensional pattern* $T$, *then* $r(F) > LCR(T)$.

Local checkability is a very general necessary condition for solvability.
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** If $F$ is a robust solution to 1-dimensional any pattern $T$, then

$$r(F) > LCR(T).$$

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** *If F is a robust solution to 1-dimensional any pattern T, then*

\[ r(F) > \text{LCR}(T) \]

**Proposition.** *If F is a robust solution to 1-dimensional pattern T, then*

\[ r(F) > \text{LCR}(T) \]

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- **Sharp existence condition:** 
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** If $F$ is a robust solution to 1-dimensional any pattern $T$, then $r(F) \geq \text{LCR}(T)$.

**Proposition.** If $F$ is a robust solution to 1-dimensional pattern $T$, then $r(F) \geq \text{LCR}(T)$.

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- Sharp existence condition: $\text{LCR}(T) = \infty$ means unsolvability
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** *If* \( F \) *is a robust solution to 1-dimensional any pattern* \( T \), *then*

\[
r(F) \leq LCR(T)
\]

**Proposition.** *If* \( F \) *is a robust solution to 1-dimensional pattern* \( T \), *then*

\[
r(F) \geq LCR(T)
\]

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- Sharp existence condition: \( LCR(T) = \infty \) means unsolvability
- and a resource condition:
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** If $F$ is a robust solution to 1-dimensional any pattern $T$, then $r(F) > LCR(T)$.

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- **Sharp existence condition:** $LCR(T) = \infty$ means unsolvability
- **and a resource condition:** $LCR(T)$ is a lower bound.
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** *If* \( F \) *is a robust solution to 1-dimensional any pattern* \( T \), *then* \( r(F) \geq LCR(T) \).

**Proposition.** *If* \( F \) *is a robust solution to 1-dimensional pattern* \( T \), *then* \( r(F) \geq LCR(T) \).

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- **Sharp existence condition:** \( LCR(T) = \infty \) means unsolvability
- **and a resource condition:** \( LCR(T) \) is a lower bound.

**Obvious next questions:**
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** If $F$ is a robust solution to 1-dimensional any pattern $T$, then

$$r(F) > LCR(T)$$

Local checkability is a very general necessary condition for solvability.

The proposition yields both a:

- Sharp existence condition: $LCR(T) = \infty$ means unsolvability
- and a resource condition: $LCR(T)$ is a lower bound.

Obvious next questions: 1) What kinds of patterns are locally checkable?
Local Checkability

We (essentially) have seen that a 1-D pattern must be locally checkable for there to be a robust solution: But actually:

**Proposition.** *If $F$ is a robust solution to 1-dimensional any pattern $T$, then $r(F) \geq LCR(T)$.*

Local checkability is a very general necessary condition for solvability. The proposition yields both a:

- Sharp existence condition: $LCR(T) = \infty$ means unsolvability
- and a resource condition: $LCR(T)$ is a lower bound.

Obvious next questions: 1) What kinds of patterns are locally checkable? And: 2) When is Local Checkability sufficient? Can we obtain sufficiency by making generic constructions?
Local Checkability
Local Checkability

Q: What kinds of patterns are locally checkable?
Local Checkability

Q: What kinds of patterns are locally checkable?

• All repeat patterns are locally checkable.
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,

\[ T_{100} = \{(100)^n\} \]

has check radius 1,
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,
  \[ T_{100} = \{(100)^n\} \]
  has check radius 1, while
  \[ T_{1000000} = \{(1000000)^n\} \]
  has check radius 3,
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance, 
  \[ T_{100} = \{(100)^n\} \]
  has check radius 1, while 
  \[ T_{1000000} = \{(1000000)^n\} \]
  has check radius 3, and 
  \[ T_{100112001} = \{(100112001)^n\} \]
  has check radius 2.
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,
  \[ T_{100112001} = \{(100112001)^n\} \]

has check radius 2. While
Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,

\[ T_{100112001} = \{(100112001)^n\} \]

has check radius 2. while

has check radius 2.
Local Checkability

Q: What kinds of patterns are locally checkable?

• All repeat patterns are locally checkable. For instance,

\[ T_{100112001} = \{(100112001)^n\} \]

has check radius 2. while

\[
\text{In fact, whenever “repeat” is defined,}
\]

has check radius 2.
Local Checkability

Q: What kinds of patterns are locally checkable?

- All repeat patterns are locally checkable. For instance,

\[ T_{100112001} = \{(100112001)^n\} \]

has check radius 2. while

In fact, whenever “repeat” is defined,

\[ LCR(T_q) \leq \frac{|q|}{2} \]

where \( q \) is the unit being repeated.
Local Checkability
Local Checkability

**Definition.** $T$ is locally generated if $T = T_\Theta$ for some $\Theta$. 
Local Checkability

**Definition.** $T$ is locally generated if $T = T_\Theta$ for some $\Theta$.

- Locally generated patterns are closed under logical ‘AND’:
Local Checkability

**Definition.** *T* is locally generated *if* $T = T_{\Theta}$ *for some* $\Theta$.

- Locally *generated* patterns are closed under logical ‘AND’:

  $$\Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2$$
Local Checkability

**Definition.** $T$ is locally generated if $T = T_{\Theta}$ for some $\Theta$.

- Locally generated patterns are closed under logical ‘AND’:
  $\Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2$

- and ‘OR’
Local Checkability

**Definition.** $T$ is locally generated if $T = T_\Theta$ for some $\Theta$.

- Locally generated patterns are closed under logical ‘AND’:
  $$\Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2$$

- and ‘OR’
  $$\Theta_1 \lor \Theta_1 \mapsto (\Theta_1 \cdot \Theta_2 + \Theta_1 + \Theta_2) \mod 2$$
Local Checkability

**Definition.** *T* is locally generated if $T = T_\Theta$ for some $\Theta$.

- Locally generated patterns are closed under logical ‘AND’:
  $$\Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2$$

- and ‘OR’
  $$\Theta_1 \lor \Theta_1 \mapsto (\Theta_1 \cdot \Theta_2 + \Theta_1 + \Theta_2) \mod 2$$

so

$$LCR(\Theta_1 \land, \lor \Theta_1) \leq \max(LCR(\Theta_1), LCR(\Theta_2))$$
Local Checkability

**Definition.** $T$ is locally generated if $T = T_{\Theta}$ for some $\Theta$.

- Locally generated patterns are closed under logical ‘AND’:
  \[ \Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2 \]

- and ‘OR’
  \[ \Theta_1 \lor \Theta_1 \mapsto (\Theta_1 \cdot \Theta_2 + \Theta_1 + \Theta_2) \mod 2 \]
  so
  \[ LCR(\Theta_1 \land, \lor \Theta_1) \leq \max(LCR(\Theta_1), LCR(\Theta_2)) \]

- and weakly closed under logical ‘NOT’, i.e.
Local Checkability

**Definition.** \( T \) is locally generated if \( T = T_\Theta \) for some \( \Theta \).

- Locally generated patterns are closed under logical ‘AND’:
  \[
  \Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2
  \]

- and ‘OR’
  \[
  \Theta_1 \lor \Theta_1 \mapsto (\Theta_1 \cdot \Theta_2 + \Theta_1 + \Theta_2) \mod 2
  \]
  so
  \[
  LCR(\Theta_1 \land, \lor \Theta_1) \leq max(LCR(\Theta_1), LCR(\Theta_2))
  \]

- and weakly closed under logical ‘NOT’, i.e. the pattern generated by \( \lnot \Theta \) is locally checkable.
  \[
  LCR(\lnot \Theta) \leq 2LCR(\Theta) + 1
  \]
Local Checkability

**Definition.** *T* is locally generated if \( T = T_\Theta \) for some \( \Theta \).

- Locally generated patterns are closed under logical ‘AND’:
  \[ \Theta_1 \land \Theta_2 \mapsto \Theta_1 \cdot \Theta_2 \]

- and ‘OR’
  \[ \Theta_1 \lor \Theta_1 \mapsto (\Theta_1 \cdot \Theta_2 + \Theta_1 + \Theta_2) \pmod{2} \]
  so
  \[ LCR(\Theta_1 \land, \lor \Theta_1) \leq \max(LCR(\Theta_1), LCR(\Theta_2)) \]

- and weakly closed under logical ‘NOT’, i.e. the pattern generated by \( \neg \Theta \) is locally checkable.
  \[ LCR(\neg \Theta) \leq 2 LCR(\Theta) + 1 \]

- Hence,
  \[ LCR(\varphi) \leq 2^{\text{rank}(\varphi)+1} \]
Local Checkability
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

• In 1-D, no nontrivial proportionate pattern is LC’able.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

- In 1-D, no nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

- In 1-D, no nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for

\[ T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \} \]
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

- In 1-D, no nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for

\[ T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \} \]

we have

\[ LCR(T_1 \cdot T_2) \leq LCR(T_1) + LCR(T_2) \]
Local Checkability

Q: What kinds of patterns are locally checkable?  \textbf{Specific to 1-D.}

- In 1-D, \textbf{no} nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for
  \[ T_1 \cdot T_2 = \{ x \cdot y \mid x \in T_1, y \in T_2 \} \]
  we have
  \[ LCR(T_1 \cdot T_2) \leq LCR(T_1) + LCR(T_2) \]

For example,
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to 1-D.

- In 1-D, no nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for

$$T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \}$$

we have

$$LCR(T_1 \cdot T_2) \leq LCR(T_1) + LCR(T_2)$$

For example,

$$T_{100} \cdot T_{1000} = \{(100)^n(1000)^m | n, m \geq 1\}$$

has a radius 3 check scheme.
Local Checkability

Q: What kinds of patterns are locally checkable? (Specific to 1-D.)

- In 1-D, no nontrivial proportionate pattern is LC’able.
- In 1-D, LC’ability is closed under various concatenations, e.g. for
  \[ T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \} \]
  we have
  \[ \text{LCR}(T_1 \cdot T_2) \leq \text{LCR}(T_1) + \text{LCR}(T_2) \]
  For example,
  \[ T_{100} \cdot T_{1000} = \{ (100)^n (1000)^m | n, m \geq 1 \} \]
  has a radius 3 check scheme.

- 1-D check schemes related to formal languages, since as a result of the closure properties:
Local Checkability

Q: What kinds of patterns are locally checkable?  

Specific to 1-D.

• In 1-D, no nontrivial proportionate pattern is LC’able.
• In 1-D, LC’ability is closed under various concatenations, e.g. for
  \[ T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \} \]
  we have
  \[ LCR(T_1 \cdot T_2) \leq LCR(T_1) + LCR(T_2) \]

For example,

\[ T_{100} \cdot T_{1000} = \{ (100)^n (1000)^m | n, m \geq 1 \} \]

has a radius 3 check scheme.

• 1-D check schemes related to formal languages, since as a result of the closure properties:

**Proposition.** All locally generated 1-D patterns are regular languages, and all regular languages are locally checkable.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

• In 1-D, no nontrivial proportionate pattern is LC’able.
• In 1-D, LC’ability is closed under various concatenations, e.g. for
  \[ T_1 \cdot T_2 = \{ x \cdot y | x \in T_1, y \in T_2 \} \]
  we have
  \[ LCR(T_1 \cdot T_2) \leq LCR(T_1) + LCR(T_2) \]
  For example,
  \[ T_{100} \cdot T_{1000} = \{ (100)^n (1000)^m | n, m \geq 1 \} \]
  has a radius 3 check scheme.

• 1-D check schemes related to formal languages, since as a result of the closure properties:

  **Proposition.** All locally generated 1-D patterns are regular languages, and all regular languages are locally checkable.

... so all 1-D check schemes are combinations of things with periodicities
Local Checkability
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.

The $r = 3$ pattern on the bottom
Local Checkability

Q: What kinds of patterns are locally checkable?  

Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.

The $r = 3$ pattern on the bottom is the $(1,4)$ encoding of the $r = 1$ pattern on the top.
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to 1-D.

Definition. A pattern $T$ over state set $S$ is $(r,m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.

The $r = 3$ pattern on the bottom is the $(1,4)$ encoding of the $r = 1$ pattern on the top.

We’ll come back to this radius/state “tradeoff”, but ...
Local Checkability

Q: What kinds of patterns are locally checkable?  

Specific to 1-D.

**Definition.** A pattern $T$ over state set $S$ is $(r, m)$-locally encodable if it can be generated by applying a radius-$r$ local rule once (synchronously) to a radius-$r$ locally checkable pattern over $m$ states.

The $r = 3$ pattern on the bottom is the $(1,4)$ encoding of the $r = 1$ pattern on the top.

We’ll come back to this radius/state “tradeoff”, but ...

**Proposition.** In 1-D, all local encodings of locally checkable patterns are again locally checkable.
Local Checkability
Local Checkability

Q: What kinds of patterns are locally checkable?  Specific to higher-D.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

• In 1-D, all patterns had combinations of periodic structures, but in higher D there can be irreducible aperiodicites.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

- In 1-D, all patterns had combinations of periodic structures, but in higher D there can be irreducible aperiodicities.

The Sierpinski Gasket has a radius-one check scheme.
Local Checkability
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

• In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.
Local Checkability

Q: What kinds of patterns are locally checkable?  

- In 1-D, **no** nontrivial proportionate pattern are LC, but in higher D they **all** essentially are.

Specific to higher-D.
Local Checkability

Q: What kinds of patterns are locally checkable?  

• In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.

Specific to higher-D.

The Cross Pattern
(r = 1, m = 2)
Local Checkability

Q: What kinds of patterns are locally checkable?  

Specific to higher-D.

- In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.

Center-Marked Pattern  
(r = 1, m = 3)
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

• In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.

Axis pattern
(r = 2, m = 3)
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

• In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

- In 1-D, no nontrivial proportionate pattern are LC, but in higher D they all essentially are.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

- Quadratic splines (ellipsoids) and cubic splines are also locally encodable.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

- Quadratic splines (ellipsoids) and cubic splines are also locally encodable.
Local Checkability

Q: What kinds of patterns are locally checkable? Specific to higher-D.

• Quadratic splines (ellipsoids) and cubic splines are also locally encodable.

So, in effect, a vector pattern language is available in regular structures above 1 dimension.
Local Checkability
Local Checkability

Variety of ways to characterize LCSs:
Local Checkability

Variety of ways to characterize LCSs:

• As “part lists” or “tile sets”:
Local Checkability

Variety of ways to characterize LCSs:

- As “part lists” or “tile sets”:
  
  Since
Local Checkability

Variety of ways to characterize LCSs:

- As “part lists” or “tile sets”:

  Since \( \Theta : B_r \rightarrow \{0, 1\} \)
Local Checkability

Variety of ways to characterize LCSs:

- As “part lists” or “tile sets”:
  
  Since \( \Theta : B_r \rightarrow \{0, 1\} \),

\( B_r \) represents the boundary of the region of interest.
Local Checkability

Variety of ways to characterize LCSs:

• As “part lists” or “tile sets”:

  Since \( \Theta : \mathcal{B}_r \rightarrow \{0, 1\} \), \( \Theta^{-1}(1) \subseteq \mathcal{B}_r \)
Local Checkability

Variety of ways to characterize LCSs:

• As “part lists” or “tile sets”:

Since \( \Theta : \mathcal{B}_r \rightarrow \{0, 1\} \), \( \Theta^{-1}(1) \subset \mathcal{B}_r \)

These are the “accepted local parts” which “fit together” to form local steady states.
Local Checkability

Variety of ways to characterize LCSs:

- As “part lists” or “tile sets”:
  
  Since  \( \Theta : B_r \rightarrow \{0, 1\} \),  \( \Theta^{-1}(1) \subset B_r \)
  
  These are the “accepted local parts” which “fit together” to form local steady states.

- In 1-D, as Formal Languages, but harder for higher dim.
Local Checkability

Variety of ways to characterize LCSs:

- As “part lists” or “tile sets”:

  Since \( \Theta : B_r \rightarrow \{0, 1\} \), \( \Theta^{-1}(1) \subset B_r \)

  These are the “accepted local parts” which “fit together” to form local steady states.

- In 1-D, as Formal Languages, but harder for higher dim.

- **Graph-theoretically.**
Local Check Schemes as Graphs

**Definition.** Given an underlying geometry $\mathcal{G}$ and label set $S$, the length-$n$ shift graph over $\mathcal{G}$, $S$ is the derived graph

$$\mathcal{D}_n(\mathcal{G}, S) = (V, E)$$

where

$$V = \{\text{diameter-n induced subgraphs in } S \text{-configurations over } \mathcal{G}\}$$

taken up to graph isomorphism, and where

$$(u, v) \in E \iff v \text{ is a 1-shift of } u.$$
Local Check Schemes as Graphs

**Definition.** Given an underlying geometry $\mathcal{G}$ and label set $S$, the length-$n$ shift graph over $\mathcal{G}, S$ is the derived graph

$$D_n(\mathcal{G}, S) = (V, E)$$

where

$$V = \{\text{diameter-}n \text{ induced subgraphs in } S \text{-configurations over } \mathcal{G} \}$$

taken up to graph isomorphism, and where

$$(u, v) \in E \iff v \text{ is a } 1\text{-shift of } u.$$
Local Check Schemes as Graphs

**Definition.** Given an underlying geometry $\mathcal{G}$ and label set $S$, the length-$n$ shift graph over $\mathcal{G}$, $S$ is the derived graph

$$\mathcal{D}_n(\mathcal{G}, S) = (V, E)$$

where

$$V = \{\text{diameter-}n \text{ induced subgraphs in } S\text{-} configurations \text{ over } \mathcal{G}\}$$

taken up to graph isomorphism, and where

$$(u, v) \in E \iff v \text{ is a 1-shift of } u.$$
Local Check Schemes as Graphs

**Definition.** Given an underlying geometry $\mathcal{G}$ and label set $S$, the length-$n$ shift graph over $\mathcal{G}$, $S$ is the derived graph

$$\mathcal{D}_n(\mathcal{G}, S) = (V, E)$$

where

$$V = \{\text{diameter}-n \text{ induced subgraphs in } S \text{-configurations over } \mathcal{G}\}$$

taken up to graph isomorphism, and where

$$(u, v) \in E \iff v \text{ is a 1-shift of } u.$$

$v$ is a 1-shift of $u$ if there is a configuration $X$ and agents $x, y \in X$ such that $\text{dist}(x, y) = 1$ and $B_r(x, X) = u, B_r(y, X) = v$. 

Thursday, November 28, 13
Local Check Schemes as Graphs

**Definition.** Given an underlying geometry $\mathcal{G}$ and label set $S$, the length-$n$ shift graph over $\mathcal{G}, S$ is the derived graph

$$\mathcal{D}_n(\mathcal{G}, S) = (V, E)$$

where

$$V = \{\text{diameter-}n \text{ induced subgraphs in } S \text{-configurations over } \mathcal{G}\}$$

taken up to graph isomorphism, and where

$$(u, v) \in E \iff v \text{ is a } 1\text{-shift of } u.$$
Local Check Schemes as Graphs
Local Check Schemes as Graphs

**Proposition.** Radius-\(r\) local check schemes over \(G, S\), are in 1-1 correspondence with subgraphs of \(\mathcal{D}_{2r+1}(G, S)\).
Local Check Schemes as Graphs

**Proposition.** Radius-\(r\) local check schemes over \(G, S\), are in 1-1 correspondence with subgraphs of \(\mathcal{D}_{2r+1}(G, S)\).

The reason why is:
Local Check Schemes as Graphs

**Proposition.** Radius-\( r \) local check schemes over \( \mathcal{G}, S \), are in 1-1 correspondence with subgraphs of \( D_{2r+1}(\mathcal{G}, S) \).

The reason why is:

\[ \Theta \longrightarrow \Theta^{-1}(1) \]
Local Check Schemes as Graphs

**Proposition.** Radius-\( r \) local check schemes over \( \mathcal{G}, S \), are in 1-1 correspondence with subgraphs of \( \mathcal{D}_{2r+1}(\mathcal{G}, S) \).

The reason why is:

\[ \Theta \rightarrow \Theta^{-1}(1) \]

And balls of radius \( r \) have diameter \( 2r+1 \), so \( \Theta^{-1}(1) \) is a subset of the nodes of \( \mathcal{D}_{2r+1}(\mathcal{G}, S) \). So
Local Check Schemes as Graphs

**Proposition.** Radius-$r$ local check schemes over $G$, $S$, are in 1-1 correspondence with subgraphs of $D_{2r+1}(G, S)$.

The reason why is:

$$\Theta \rightarrow \Theta^{-1}(1)$$

And balls of radius $r$ have diameter $2r+1$, so $\Theta^{-1}(1)$ is a subset of the nodes of $D_{2r+1}(G, S)$. So

$$\Theta \rightarrow \Theta^{-1}(1) \rightarrow \text{induced subgraph } G(\Theta)$$
Proposition. Radius-\(r\) local check schemes over \(G, S\), are in 1-1 correspondence with subgraphs of \(D_{2r+1}(G, S)\).

The reason why is:

\[ \Theta \rightarrow \Theta^{-1}(1) \]

And balls of radius \(r\) have diameter \(2r+1\), so \(\Theta^{-1}(1)\) is a subset of the nodes of \(D_{2r+1}(G, S)\). So

\[ \Theta \rightarrow \Theta^{-1}(1) \rightarrow \text{induced subgraph } G(\Theta) \]

In words: local check schemes are equivalent to graphs, and in fact subgraphs of a very specific “ambient space.”
Local Check Schemes as Graphs

**Proposition.** Radius-\(r\) local check schemes over \(G, S\), are in 1-1 correspondence with subgraphs of \(D_{2r+1}(G, S)\).

The reason why is:

\[
\Theta \rightarrow \Theta^{-1}(1)
\]

And balls of radius \(r\) have diameter \(2r+1\), so \(\Theta^{-1}(1)\) is a subset of the nodes of \(D_{2r+1}(G, S)\). So

\[
\Theta \rightarrow \Theta^{-1}(1) \rightarrow \text{induced subgraph } G(\Theta)
\]

In words: local check schemes are equivalent to graphs, and in fact subgraphs of a very specific “ambient space.”

\(D_n(\mathbb{Z}, 2)\) is known (from other contexts) as the DeBruijn graph, so the generalized DeBruijn graphs are the “ambient spaces” of locally checkable patterns.
Local Check Schemes as Graphs
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

\[ T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\} \]
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

$$T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\}$$

is associated with the graph
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

\[ T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\} \]

is associated with the graph
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

\[ T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\} \]

is associated with the graph

The pattern
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

\[ T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\} \]

is associated with the graph

\[ T_{100} \cdot T_{1000} = \{(100)^n(1000)^m | n, m \geq 1\} \]
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

$$T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\}$$

is associated with the graph

The pattern

$$T_{100} \cdot T_{1000} = \{(100)^n(1000)^m|n, m \geq 1\}$$

has a radius-3 check scheme whose graph is:
Local Check Schemes as Graphs

For example, the radius-2 check scheme for repeat pattern:

\[ T_{1000} = \{1000, 10001000, \ldots, (1000)^n, \ldots\} \]

is associated with the graph

The pattern

\[ T_{100} \cdot T_{1000} = \{(100)^n(1000)^m | n, m \geq 1\} \]

has a radius-3 check scheme whose graph is:
Local Check Schemes as Graphs
Local Check Schemes as Graphs
Local Check Schemes as Graphs
Local Check Schemes as Graphs

(5 x 5 torus)
Local Check Schemes as Graphs

(5 x 5 torus)
Local Check Schemes as Graphs

(5 x 5 torus)
Local Check Schemes as Graphs

(5 x 5 torus)