Characterizing the Complexity of Curricular Patterns in Engineering Programs

Gregory L. Heileman  Ahmad Slim
Michael Hickman  Chaouki T. Abdallah
{heileman, ahslim, mhickman, chaouki}@unm.edu

Department of Electrical & Computer Engineering
University of New Mexico

June 26, 2017
Two EE Programs

Term 1
- Intro. to Eng.
- Calculus I
- Composition I
- Programming I
- Chemistry I

Term 2
- Calculus II
- Physics I
- Physics I Lab

Term 3
- Circuits I
- Physics II
- Electromagnetics
- Microprocessors

Term 4
- Technical Writing
- Signals & Systems
- Circuits II
- Electrons

Term 5
- Engineering Statistics
- Electives
- Electives
- Electives

Term 6
- US Government
- Art
- Electives
- Electives

Term 7
- Senior Design I
- Electives
- Electives
- Electives

Term 8
- Senior Design II
- Electives
- Electives
- Electives

University of New Mexico
Curricular Patterns
June 26, 2017
Two EE Programs

- American History I
- Composition I
- Intro to Eng.
- Calculus I
- Chemistry I
- College Experience
- Intro. to Eng.
- Composition I
- Programming I
- Circuit I
- Physics I
- Digital Design
- Physics II Lab
- Eng. Math
- Humanities 1
- Humanities 2
- HW Design Lang.
- Physics III
- Signals & Systems I
- Computer Organization
- Computer Organization
- Linear Algebra
- Random Signals

- American History II
- Composition II
- Programming I
- Calculus II
- Physics I Lab
- Eng. Math
- Physics II
- Composition II
- Programming II
- Signals & Systems
- Electromagnetics
- Microprocessors
- Electives

- State Government
- Technical Writing
- Art
- US Government
- French
- English
- Humanities 2
- Electives

- Circuit II
- Signals & Systems
- Electromagnetics
- Microprocessors
- Electives

- Physics I Lab
- Engineering Statistics
- Electives

- Circuit Lab
- Electives

- Physics II Lab
- Electives

- Physics II
- Electives

- Engineering Statistics
- Electives

- Circuit Lab
- Electives

- Electives

- Circuit I
- Signals & Systems
- Electromagnetics
- Microprocessors
- Electives

- Circuit Lab
- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives

- Electives
Two EE Programs

Term 1
- Intro to Eng.
- Calculus I
- Chemistry I
- Composition I
- Physics I Lab
- Circuits I

Term 2
- Composition II
- Calculus II
- Programming I
- Eng. Mech
- Physics II Lab

Term 3
- Calculus III
- Signals & System
- Circuits II
- Electromagnetics
- Eng. Electromag.

Term 4
- Programming II
- Electronics
- Circuits Lab
- Elective
- Humanities II

Term 5
- Physics I Lab
- Signals & System
- Circuits Lab
- Elective
- Humanities II

Term 6
- Physics II Lab
- Electromagnetics
- Elective
- Elective
- Tech. Elec. II

Term 7
- Physics III
- HW Design Lang.
- Elective
- Elective
- Tech. Elec. III

Term 8
- Circuits Lab
- Elective
- Elective
- Elective
- Tech. Elec. IV

University of New Mexico

Curricular Patterns

June 26, 2017
We represent a curriculum $C$ consisting of $n$ courses as a directed graph $G_C = (V, E)$,
We represent a curriculum $C$ consisting of $n$ courses as a directed graph $G_C = (V, E)$, where:

- **Courses**: each vertex $v_1, \ldots, v_n \in V$ represents a requirement in $C$, 
- **Prerequisites**: there is a directed edge $(v_i, v_j) \in E$ from requirement $v_i$ to $v_j$ if $v_i$ must be satisfied prior to the satisfaction of $v_j$. 

We refer to $G_C$ as a curriculum graph. The structure of $G_C$ influences how difficult it is to complete a curriculum.
We represent a curriculum $C$ consisting of $n$ courses as a directed graph $G_C = (V, E)$, where:

- **Courses**: each vertex $v_1, \ldots, v_n \in V$ represents a requirement in $C$,

- **Prerequisites**: there is a directed edge $(v_i, v_j) \in E$ from requirement $v_i$ to $v_j$ if $v_i$ must be satisfied prior to the satisfaction of $v_j$. 

We refer to $G_C$ as a curriculum graph. The structure of $G_C$ influences how difficult it is to complete a curriculum. The structural complexity of $C$, denoted $\alpha_C$, is a function of relevant properties of $G_C$: $\alpha_C = g(G_C)$. 

University of New Mexico
Curricular Patterns
June 26, 2017 3 / 25
We represent a curriculum $C$ consisting of $n$ courses as a directed graph $G_C = (V, E)$, where:

- **Courses**: each vertex $v_1, \ldots, v_n \in V$ represents a requirement in $C$,

- **Prerequisites**: there is a directed edge $(v_i, v_j) \in E$ from requirement $v_i$ to $v_j$ if $v_i$ must be satisfied prior to the satisfaction of $v_j$.

We refer to $G_C$ as a **curriculum graph**.
We represent a curriculum $C$ consisting of $n$ courses as a directed graph $G_C = (V, E)$, where:

- **Courses**: each vertex $v_1, \ldots, v_n \in V$ represents a requirement in $C$,

- **Prerequisites**: there is a directed edge $(v_i, v_j) \in E$ from requirement $v_i$ to $v_j$ if $v_i$ must be satisfied prior to the satisfaction of $v_j$.

We refer to $G_C$ as a **curriculum graph**. The structure of $G_C$ influences how difficult it is to complete a curriculum.
We represent a curriculum \( C \) consisting of \( n \) courses as a directed graph \( G_C = (V, E) \), where:

- **Courses**: each vertex \( v_1, \ldots, v_n \in V \) represents a requirement in \( C \),

- **Prerequisites**: there is a directed edge \( (v_i, v_j) \in E \) from requirement \( v_i \) to \( v_j \) if \( v_i \) must be satisfied prior to the satisfaction of \( v_j \).

We refer to \( G_C \) as a **curriculum graph**. The structure of \( G_C \) influences how difficult it is to complete a curriculum.

The **structural complexity** of \( C \), denoted \( \alpha_C \), is a function of relevant properties of \( G_C \):

\[
\alpha_C = g(G_C).
\]
Structural complexity is completely determined by $G_C$. 
Structural Complexity

- Structural complexity is completely determined by $G_C$.

- The following graph properties impact student progression:
  - **Delay Factor**: characterized by long paths in the curriculum.
Structural complexity is completely determined by $G_C$.

The following graph properties impact student progression:

- **Delay Factor**: characterized by long paths in the curriculum.

- **Blocking Factor**: the number of courses a student is precluded from taking until they pass a given class.
Structural complexity is completely determined by $G_C$.

The following graph properties impact student progression:

- **Delay Factor**: characterized by long paths in the curriculum.

- **Blocking Factor**: the number of courses a student is precluded from taking until they pass a given class.

- **Central Courses**: key courses in a curriculum — many prerequisites must be satisfied to reach them, and they “unblock” many courses in the curriculum that follow them.
Structural Complexity

- Structural complexity is completely determined by $G_C$.

- The following graph properties impact student progression:
  - **Delay Factor**: characterized by long paths in the curriculum.
  - **Blocking Factor**: the number of courses a student is precluded from taking until they pass a given class.
  - **Central Courses**: key courses in a curriculum — many prerequisites must be satisfied to reach them, and they “unblock” many courses in the curriculum that follow them.
  - **Degrees of Freedom**: the extent to which a curriculum can be rearranged if certain courses are not passed.
The **delay factor** associated with course \( v_k \) in curriculum \( G_C = (V, E) \), denoted \( d(v_k) \), is the number of nodes in the longest path in \( G_C \) that passes through \( v_k \).
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:
The **delay factor** associated with course \( v_k \) in curriculum \( G_C = (V, E) \), denoted \( d(v_k) \), is the number of nodes in the longest path in \( G_C \) that passes through \( v_k \).

- \( t = \max_{v_k \in V} d(v_k) \) determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least \( t + 1 \) terms will be required to complete the curriculum.

**Ex:**

![Diagram of a directed graph with nodes labeled 1, 2, 3, 4, and 5 connected by directed edges.]
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:

\[ C_1: \]

\[ \begin{array}{c}
V_1 & \rightarrow & V_2 & \rightarrow & V_3 \\
3 & \rightarrow & & \rightarrow & \\
V_4 & \rightarrow & \\
3 & \rightarrow & & \rightarrow & \\
\end{array} \]
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

Exhibit:
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

![Diagram](image)
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

![Diagram](image)
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

```
C_1:
  \[ V_1 \rightarrow V_2 \rightarrow V_3 \]
  \[ V_4 \]
```
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

```
C_1: 3 → 3 → 3

v_1 → v_2 → v_3

v_4 → 2
```
The **delay factor** associated with course \( v_k \) in curriculum \( G_C = (V, E) \), denoted \( d(v_k) \), is the number of nodes in the longest path in \( G_C \) that passes through \( v_k \).

- \( t = \max_{v_k \in V} d(v_k) \) determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least \( t + 1 \) terms will be required to complete the curriculum.

**Ex:**

![Diagram](image)
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:
The delay factor associated with course \( v_k \) in curriculum \( G_C = (V, E) \), denoted \( d(v_k) \), is the number of nodes in the longest path in \( G_C \) that passes through \( v_k \).

- \( t = \max_{v_k \in V} d(v_k) \) determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least \( t + 1 \) terms will be required to complete the curriculum.

Ex:
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

![Graph C1](image1)

$C_1: \quad \begin{array}{c}
3 \\
\downarrow \\
3 \\
\downarrow \\
3
\end{array}$

$C_2: \quad \begin{array}{c}
2 \\
\downarrow \\
V_1 \\
\downarrow \\
V_4 \\
\downarrow \\
V_3
\end{array}$
The delay factor associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

Ex:
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

$$C_1: \quad \begin{array}{ccc}
V_1 & \rightarrow & V_2 \\
3 & \rightarrow & 3 \\
& \rightarrow & 3 \\
V_4 & \rightarrow & 2 \\
2 & \rightarrow & 2
\end{array}$$

$$C_2: \quad \begin{array}{ccc}
V_1 & \rightarrow & V_2 \\
2 & \rightarrow & 2 \\
& \rightarrow & 2 \\
V_3 & \rightarrow & 2 \\
V_4 & \rightarrow & 2
\end{array}$$
The **delay factor** associated with course $v_k$ in curriculum $G_C = (V, E)$, denoted $d(v_k)$, is the number of nodes in the longest path in $G_C$ that passes through $v_k$.

- $t = \max_{v_k \in V} d(v_k)$ determines the minimum number of terms required to complete a curriculum.
- Failure to complete a course on the longest path means that at least $t + 1$ terms will be required to complete the curriculum.

**Ex:**

![Diagram](attachment:image.png)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightarrow v_j; \\
0, & \text{if } v_i \not\rightarrow v_j.
\end{cases}$$
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} l(v_i, v_j),$$

where $l$ is the indicator function:

$$l(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightsquigarrow v_j; \\
0, & \text{if } v_i \not\rightsquigarrow v_j.
\end{cases}$$

**Ex:**

![Diagram](image.png)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightsquigarrow v_j; \\
0, & \text{if } v_i \not\rightsquigarrow v_j.
\end{cases}$$

**Ex:**

![Diagram](image)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightarrow v_j; \\
0, & \text{if } v_i \not\rightarrow v_j.
\end{cases}$$

**Ex:**

```
C_1:
\begin{align*}
&v_1 \rightarrow v_2 \\
&v_4 \rightarrow v_1
\end{align*}
```
Define the **blocking factor** associated with course \( v_i \), denoted \( b(v_k) \) in curriculum \( G_C = (V, E) \) as:

\[
b(v_i) = \sum_{v_j \in V} I(v_i, v_j),
\]

where \( I \) is the indicator function:

\[
I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightarrow v_j; \\
0, & \text{if } v_i \not\rightarrow v_j.
\end{cases}
\]

**Ex:**

![Diagram of the blocking factor](image)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \leadsto v_j; \\
0, & \text{if } v_i \not\leadsto v_j.
\end{cases}$$

**Ex:**

![Diagram of a network with nodes $v_1, v_2, v_3, v_4$ and edges illustrating blocking relationships.]
Define the blocking factor associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 1, & \text{if } v_i \rightarrow v_j; \\ 0, & \text{if } v_i \not\rightarrow v_j. \end{cases}$$

Ex:

![Diagram of courses and relationships]

$C_1$: $v_1$ → $v_2$ → $v_3$ → $v_4$
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} l(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \implies v_j; \\
0, & \text{if } v_i \not\implies v_j.
\end{cases}$$

**Ex:**

![Graph](image-url)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightarrow v_j; \\
0, & \text{if } v_i \nrightarrow v_j.
\end{cases}$$

**Ex:**

![Diagram](image)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 1, & \text{if } v_i \xrightarrow{} v_j; \\ 0, & \text{if } v_i \xrightarrow{} v_j. \end{cases}$$

**Ex:**

$$C_1: \begin{array}{ccc} V_1 & \rightarrow & V_2 \\ \downarrow & & \downarrow \\ 3 & \rightarrow & 1 \\ \downarrow & & \downarrow \\ V_4 & \rightarrow & 0 \end{array} \quad C_2: \begin{array}{ccc} V_1 & \rightarrow & V_2 \\ \downarrow & & \downarrow \\ 3 & \rightarrow & V_3 \\ \downarrow & & \downarrow \\ V_4 \end{array}$$
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \twoheadrightarrow v_j; \\
0, & \text{if } v_i \not\twoheadrightarrow v_j.
\end{cases}$$

**Ex:**

![Diagram](image)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightsquigarrow v_j; \\
0, & \text{if } v_i \not\rightsquigarrow v_j.
\end{cases}$$

**Ex:**

![Diagram](image)
Define the **blocking factor** associated with course \( v_i \), denoted \( b(v_k) \) in curriculum \( G_C = (V, E) \) as:

\[
b(v_i) = \sum_{v_j \in V} I(v_i, v_j),
\]

where \( I \) is the indicator function:

\[
I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \rightarrow v_j; \\
0, & \text{if } v_i \not\rightarrow v_j.
\end{cases}
\]

**Ex:**

![Diagram](image)
Define the **blocking factor** associated with course $v_i$, denoted $b(v_k)$ in curriculum $G_C = (V, E)$ as:

$$b(v_i) = \sum_{v_j \in V} I(v_i, v_j),$$

where $I$ is the indicator function:

$$I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \Rightarrow v_j; \\
0, & \text{if } v_i \not\Rightarrow v_j. 
\end{cases}$$

**Ex:**

![Diagram](image-url)
Define the **blocking factor** associated with course \( v_i \), denoted \( b(v_k) \) in curriculum \( G_C = (V, E) \) as:

\[
b(v_i) = \sum_{v_j \in V} I(v_i, v_j),
\]

where \( I \) is the indicator function:

\[
I(v_i, v_j) = \begin{cases} 
1, & \text{if } v_i \xrightarrow{} v_j; \\
0, & \text{if } v_i \xrightarrow{} v_j.
\end{cases}
\]

**Ex:**

\[C_1: \quad \begin{array}{c}
3 \quad 1 \quad 0 \\
\downarrow \quad \downarrow \quad \downarrow \\
V_4 \quad V_2 \quad V_3 \\
\end{array} \quad C_2: \quad \begin{array}{c}
3 \quad 0 \\
\downarrow \quad \downarrow \\
V_4 \quad V_3 \\
\end{array} \]
Network Information Theory —

- **Betweenness Centrality**: measures the extent to which a vertex lies on paths between other vertices.

- We’re working on the proper way to quantify this.
Structural Complexity – Degrees of Freedom

\[ C_1: \]
\[ \begin{array}{c}
  V_1 \\
  V_2 \\
  V_3 \\
  V_4 \\
\end{array} \]

\[ C_2: \]
\[ \begin{array}{c}
  V_1 \\
  V_2 \\
  V_3 \\
  V_4 \\
\end{array} \]

Assume \( V_1 \) and \( V_4 \) are not passed on first attempt.
$C_1$ Plan —

**term 1:** $v_1, v_2, v_4$

**term 2:** $v_3$

$C_2$ Revised Plan —

**term 1:** $v_1, v_2, v_4$

**term 2:** $v_1, v_4$

**term 3:** $v_3$
$C_1$ Plan —
term 1: $v_1, v_2, v_4$
term 2: $v_3$

$C_2$ Plan —
term 1: $v_1, v_2$
term 2: $v_3, v_4$
Structural Complexity – Degrees of Freedom

$C_1$ Plan —
term 1: $v_1, v_2, v_4$
term 2: $v_3$

$C_2$ Plan —
term 1: $v_1, v_2$
term 2: $v_3, v_4$

Assume $v_1$ and $v_4$ are not passed on first attempt.
Structural Complexity – Degrees of Freedom

$C_1$: $v_1$ → $v_3$
$v_2$ → $v_4$

$C_2$: $v_1$ → $v_3$
$v_2$ → $v_4$

$C_1$ Plan —
- term 1: $v_1$, $v_2$, $v_4$
- term 2: $v_3$

$C_2$ Plan —
- term 1: $v_1$, $v_2$
- term 2: $v_3$, $v_4$

Assume $v_1$ and $v_4$ are not passed on first attempt.

$C_1$ Revised Plan —
- term 1: $v_1$, $v_2$, $v_4$
- term 2: $v_1$, $v_4$
- term 3: $v_3$
Structural Complexity – Degrees of Freedom

$C_1$ Plan —
term 1: $v_1, v_2, v_4$
term 2: $v_3$

$C_2$ Plan —
term 1: $v_1, v_2$
term 2: $v_3, v_4$

Assume $v_1$ and $v_4$ are not passed on first attempt.

$C_1$ Revised Plan —
term 1: $v_1, v_2, v_4$
term 2: $v_1, v_4$
term 3: $v_3$

$C_2$ Revised Plan —
term 1: $v_1, v_2$
term 2: $v_1$
term 3: $v_3, v_4$
term 4: $v_4$
The aforementioned graph properties all influence progression when students are unable to complete classes.
The aforementioned graph properties all influence progression when students are unable to complete classes. **Consider the two extremes:**

- Students always complete every class on the first attempt
The aforementioned graph properties all influence progression when students are unable to complete classes. Consider the two extremes:

- Students always complete every class on the first attempt — $G_C$ is nearly irrelevant, all students will graduate on time.

- Students are never able to complete a class — $G_C$ is irrelevant, no student will ever graduate.
The aforementioned graph properties all influence progression when students are unable to complete classes. Consider the two extremes:

- Students always complete every class on the first attempt — $G_C$ is nearly irrelevant, all students will graduate on time.
- Students are never able to complete a class
Instructional Complexity

- The aforementioned graph properties all influence progression when students are unable to complete classes. **Consider the two extremes:**
  - Students always complete every class on the first attempt — \( G_C \) is nearly irrelevant, all students will graduate on time.
  - Students are never able to complete a class — \( G_C \) is irrelevant, no student will ever graduate.

- There is another factor at play here.
Instructional Complexity

- The aforementioned graph properties all influence progression when students are unable to complete classes. 
  **Consider the two extremes:**
  - Students always complete every class on the first attempt — $G_C$ is nearly irrelevant, all students will graduate on time.
  - Students are never able to complete a class — $G_C$ is irrelevant, no student will ever graduate.

- There is another factor at play here.

- The *instructional complexity* of curriculum $C$, denoted $\gamma_C$, is a function of the difficulties of the courses in $C$:

  $$\gamma_C = h(\text{course difficulties})$$
The difficulty of a course is a function of numerous factors, including:

- instructor quality
The difficulty of a course is a function of numerous factors, including:
- instructor quality
- course content
The difficulty of a course is a function of numerous factors, including:

- instructor quality
- course content
- support services provided
The difficulty of a course is a function of numerous factors, including:

- instructor quality
- course content
- support services provided
- student background preparation
The difficulty of a course is a function of numerous factors, including:
- instructor quality
- course content
- support services provided
- student background preparation
- etc.
Instructional Complexity

- The difficulty of a course is a function of numerous factors, including:
  - instructor quality
  - course content
  - support services provided
  - student background preparation
  - etc.

- $\gamma_C$ is extremely difficult to characterize.
Instructional Complexity

- The difficulty of a course is a function of numerous factors, including:
  - instructor quality
  - course content
  - support services provided
  - student background preparation
  - etc.

- $\gamma_C$ is extremely difficult to characterize.

- The historic pass rates of the courses in a curriculum $C$ provides a good approximation to $\gamma_C$. 
The overall complexity of a curriculum $C$, denoted $\Psi_C$, is a combination of the inherent difficulty associated with traversing a curriculum graph (structural complexity), and the manner in which the courses are taught (instructional complexity):

$$\Psi_C = f(\alpha_C, \gamma_C).$$
The overall complexity of a curriculum $C$, denoted $\Psi_C$, is a combination of the inherent difficulty associated with traversing a curriculum graph (structural complexity), and the manner in which the courses are taught (instructional complexity):

$$\Psi_C = f(\alpha_C, \gamma_C).$$

$\uparrow$ complexity $\implies$ $\downarrow$ lower completion rates
The overall complexity of a curriculum $C$, denoted $\Psi_C$, is a combination of the inherent difficulty associated with traversing a curriculum graph (structural complexity), and the manner in which the courses are taught (instructional complexity):

$$\Psi_C = f(\alpha_C, \gamma_C).$$

- $\uparrow$ complexity $\implies \downarrow$ lower completion rates

- Does higher curricular complexity lead to higher quality (improved student learning outcomes)?
How do we make $\Psi_C$ useful?
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot), g(\cdot)$ and $h(\cdot)$. 
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot), g(\cdot)$ and $h(\cdot)$.

We can learn $g(\cdot)$ —

$$\Psi_C = f(\alpha_C, \gamma_C)$$
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot), g(\cdot)$ and $h(\cdot)$.

We can learn $g(\cdot)$ —

$$\Psi_C = f(\alpha_C, \gamma_C) = f(g(G_C), h(\cdot, \cdot, \cdot))$$
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot), g(\cdot)$ and $h(\cdot)$.

We can learn $g(\cdot)$ —

$$
\Psi_C = f(\alpha_C, \gamma_C)
= f(g(G_C), h(\cdot, \cdot, \cdot, \cdot))
= f(g(G_C), \text{course pass rates})
$$

—Simulation—
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot), g(\cdot)$ and $h(\cdot)$.

We can learn $g(\cdot)$ —

$$
\Psi_C = f(\alpha_C, \gamma_C) = f(g(G_C), h(\cdot, \cdot, \cdot, \cdot)) = f(g(G_C), \text{course pass rates})
$$

—Simulation—

Completion rates $= f(g(G_C), \text{fixed pass rate})$
How do we make $\Psi_C$ useful? We need to better characterize $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$.

We can learn $g(\cdot)$ —

$$
\Psi_C = f(\alpha_C, \gamma_C)
= f(g(G_C), h(\cdot, \cdot, \cdot, \cdot))
= f(g(G_C), \text{course pass rates})
$$

—Simulation—

Completion rates $= f (g(G_C), \text{fixed pass rate})$

Vary $g(\cdot)$, and measure correlation between completion rates and the various $g(\cdot)$'s.
Learning Structural Complexity

\[ g(G_C) = \sum_{v_k \in V} (d(v_k) + b(v_k)) \]
Learning Structural Complexity

\[ g(G_C) = \sum_{v_k \in V} (d(v_k) + b(v_k)) \]

\( g(G_C) \) vs. 5-term completion rates for six-course curricula balanced over 3 terms
All four-course curricula balanced over two terms:
### Structural Complexity Validation

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.5</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.5</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>50</td>
<td>75</td>
<td>87.5</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>0</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>28.13</td>
<td>50.24</td>
<td>72.30</td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>5.27</td>
<td>23.23</td>
<td>45.71</td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>3.52</td>
<td>19.25</td>
<td>41.54</td>
</tr>
</tbody>
</table>

(c)

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>3.52</td>
<td>19.25</td>
<td>41.54</td>
</tr>
</tbody>
</table>

(e)

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>3.52</td>
<td>19.25</td>
<td>41.54</td>
</tr>
</tbody>
</table>

(f)

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_2$</td>
<td>50</td>
<td>75</td>
<td>87.75</td>
<td>93.75</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>$v_4$</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>68.75</td>
</tr>
<tr>
<td>grad. rate</td>
<td>0</td>
<td>1.76</td>
<td>13.24</td>
<td>33.52</td>
</tr>
</tbody>
</table>

(g)
Application: Curricular Design Patterns
The notion of *design patterns* originated as a concept in architecture intended to capture the essence of an architectural design.
The notion of design patterns originated as a concept in architecture intended to capture the essence of an architectural design.

- Allows architects to capture, in general terms, design decisions and ideas that have proven successful in solving particular design challenges.
The notion of *design patterns* originated as a concept in architecture intended to capture the essence of an architectural design.

- Allows architects to capture, in general terms, design decisions and ideas that have proven successful in solving particular design challenges.
- Provides a useful collection of knowledge that other architects may consult in the future when confronting similar design challenges.
Architectural Design Patterns

The notion of *design patterns* originated as a concept in architecture intended to capture the essence of an architectural design.

- Allows architects to capture, in general terms, design decisions and ideas that have proven successful in solving particular design challenges.
- Provides a useful collection of knowledge that other architects may consult in the future when confronting similar design challenges.
- These design patterns constitute a language that architects may use to more efficiently communicate with one another.
Summary:

“[e]ach pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.”
In 1987, software engineers began experimenting with the notion of applying design patterns to the challenges confronted in the design of large-scale software systems.
In 1987, software engineers began experimenting with the notion of applying design patterns to the challenges confronted in the design of large-scale software systems. This led to the formalization of a large number of software design patterns that have been shown to greatly aid in the development of complex software systems.
In 1987, software engineers began experimenting with the notion of applying design patterns to the challenges confronted in the design of large-scale software systems.

This led to the formalization of a large number of software design patterns that have been shown to greatly aid in the development of complex software systems.

Provides developers with a reusable set of proven solutions to generalized problems.
In 1987, software engineers began experimenting with the notion of applying design patterns to the challenges confronted in the design of large-scale software systems.

This led to the formalization of a large number of software design patterns that have been shown to greatly aid in the development of complex software systems.

Provides developers with a reusable set of proven solutions to generalized problems.

Software design patterns are similar to their architectural counterparts in that they provide software engineers with a language that can be used to discuss software design issues.
Definition: (Curricular Design Pattern). A collection of curricular and co-curricular learning activities intentionally structured so as to allow students to attain a set of learning outcomes within a given educational context.
This curricular design pattern is constructed so as to attain a set of learning outcomes that involve the ability to design, build and analyze simple electronic circuits, under the assumption that a student is prepared for Calculus I. Some of the specific learning outcomes are as follows. Students will:

1. Understand the functions of basic electrical circuit elements and sources;

2. Have the ability to apply Ohm’s and Kirchhoff’s circuit laws in the lumped element model of electrical circuits;

3. Appreciate the consequences of linearity, in particular the principle of superposition and Thevenin and Norton equivalent circuits;

4. Understand the concept of state in a dynamical physical system and have the ability to analyze simple first and second order linear circuits containing memory elements.
A curricular pattern of courses designed to allow students to attain the learning outcomes:
A curricular pattern of courses designed to allow students to attain the learning outcomes:

- Calculus I
- Calculus II
- Calculus III
- Differential Eqs.
- Physics I
- Programming I
- Circuits I

Term 1: Calculus I

Term 2: Calculus II, Physics I

Term 3: Differential Eqs., Calculus III

Term 4: Circuits I
If we include Pre-Calculus:
If we include Pre-Calculus:

Structural Complexity = 56
Redesigning the pattern:

Structural Complexity = 41
Redesigning the pattern:

Structural Complexity = 41