Closures of relations

- Sometimes you have a relation which isn’t reflexive, or isn’t symmetric, or isn’t transitive.

- For each of these properties, we can add ordered pairs to the relation, just enough to make it have the given property. The resulting relation is called the reflexive closure, symmetric closure, or transitive closure respectively.

- Another way to say this is that for property $X$, the $X$ closure of a relation $R$ is the smallest relation containing $R$ that has property $X$, where $X$ can be “reflexive” or “symmetric” or “transitive”.

- We denote the reflexive closure of $R$ by $refc(R)$, the symmetric closure of $R$ by $symc(R)$, and the transitive closure by $tc(R)$. Another popular notation, though, for the last is $R^+$. 
Reflexive and Symmetric Closures

• These are easy, and also are not used a lot.

• **Definition** The reflexive closure of a relation $R$ on a set $A$ is defined to be $\text{refc}(R) = R \cup \text{id}_A$.

• **Example** If $R = \{(1, 2), (2, 3), (3, 2), (3, 3)\}$, then $\text{refc}(R) = \{(1, 2), (2, 3), (3, 2), (3, 3), (1, 1), (2, 2)\}$.

• **Definition** The symmetric closure of a relation $R$ on a set $A$ is defined to be $\text{symc}(R) = R \cup \bar{R}$, where $\bar{R} = \{(y, x) \mid (x, y) \in R\}$.

• **Example** If $R = \{(1, 2), (2, 3), (3, 2), (3, 3)\}$, then $\text{symc}(R) = \{(1, 2), (2, 1), (2, 3), (3, 2), (3, 3)\}$.
Transitive Closure

• This is more interesting; adding ordered pairs is an iterative process.

• Transitive closure on directed graphs shows where you can go using some number of arcs.

• To get the transitive closure, you first add all arrows that traverse (jump) two original arrows; then those that traverse three, and so forth.

• We illustrate on the next slide.
Graphical construction of transitive closure

Original Relation

Original Relation plus 2-jumps

Original Relation plus 2-jumps plus 3-jump
Why does anyone want to compute the transitive closure?

- Kevin Bacon numbers: is a given actor connected to Kevin Bacon? and if so, by how many steps?

- Actor/ess $a$ is directly connected to actor/ess $b$ if $a$ and $b$ have appeared in the same movie, as determined by the Internet Movie Database. (This is a symmetric but not transitive relation.)

- Are there any actors out there not eventually connected to Kevin Bacon?

- Check out http://www.cs.virginia.edu/oracle/
Recursive Definition of Transitive Closure

• Given a binary relation $R$ on a set $A$, we use the following rules to construct the relation $tc(R)$.

1. “Basis”: if $(x, y) \in R$, then $(x, y) \in tc(R)$.
2. “Induction”: if $(x, y) \in tc(R)$ and $(y, z) \in tc(R)$, then $(x, z) \in tc(R)$.
3. No pair is in $tc(R)$ unless it is shown there using a finite number of applications of rules 1 and 2.

• Example  Using these rules on the example on the last slide, we first use rule 1 three times to put $(a, b)$, $(b, c)$, and $(c, d)$ into $tc(R)$.

• We add the “two-jumps” with two uses of rule 2: once from $(a, b)$ and $(b, c)$ to get $(a, c)$, and then from $(b, c)$ and $(c, d)$ to get $(b, d)$.

• We then use rule 2 from $(a, c)$ and $(c, d)$ to get $(a, d)$, the three-jump.

• We need rule 3 to insure that $tc(R)$ is the smallest transitive relation containing $R$. For example, $S = A \times A$ is a much bigger transitive relation containing $R$. 
Proving that $tc(R)$ is what’s advertised

- We have to show (i) that $tc(R)$ is transitive, and (ii) that if $R \subseteq S$ and $S$ is transitive then $tc(R) \subseteq S$.
- Proving (i) is easy. If $(x, y)$ and $(y, z)$ are in $tc(R)$ then by rule 2, $(x, z)$ is in $tc(R)$.
- The proof of (ii) is a little harder. We use strong induction to show that if $(x, z) \in tc(R)$, then $(x, z) \in S$.
- The slick way to do this is to define a kind of augmented transitive closure, one where we have triples $(x, y, k)$ in it. The idea is that the natural number $k$ tells us how many rules were used to put $(x, y)$ in the transitive closure.
- We define $\hat{tc}(R)$ by structural induction. (i) if $(x, y) \in R$, then $(x, y, 1) \in \hat{tc}(R)$. (ii) If $(x, y, j)$ and $(y, z, k)$ are in $\hat{tc}(R)$, then $(x, z, j+k+1) \in \hat{tc}(R)$.
- We have that $(x, y) \in tc(R)$ if and only if for some $n$, $(x, y, n) \in \hat{tc}(R)$. 
Proof continued

Assume that $R \subseteq S$. We show by strong induction on $n$ that for all $n, x, y, (x, y, n) \in \hat{t}c(R)$ implies $(x, y) \in S$.

**Basis.** If $(x, y, 1) \in \hat{t}c(R)$, then we must have $(x, y) \in R$, so since $R \subseteq S$, we have $(x, y) \in S$, completing the basis case.

**Induction step.** Assume for all $m \leq n, u, and v$, that if $(u, v, m) \in \hat{t}c(R)$, then $(u, v) \in S$. Suppose $(x, y, n + 1) \in \hat{t}c(R)$. Then for some $w, j, k$, $(x, w, j) \in \hat{t}c(R), (w, y, k) \in \hat{t}c(R)$, and $n + 1 = j + k + 1$. Then $j$ and $k$ are both less than or equal to $n$, and by strong inductive hypothesis (used twice) we have $(x, w) \in S$ and $(w, y) \in S$. But $S$ is transitive, so $(x, y) \in S$, as we wanted.

If $(x, y) \in tc(R)$, then for some $n, (x, y, n) \in \hat{t}c(R)$. By what we just proved, we thus have $(x, y) \in S$. 
Characterizing $tc(R)$ other ways

• Since transitivity is connected to composition, it makes sense to see if there’s a way to express $tc(R)$ using composition.

• There is a formula to that effect, which leads to a matrix algorithm for calculating transitive closure.

• The formula requires defining powers of a relation inductively.

• **Definition** Let $R$ be a binary relation on $A$. For $j \geq 1$ we define the powers $R^j$ of $R$: put $R^1 = R$ and $R^{j+1} = R^j \circ R$.

• **Theorem**

\[
tc(R) = \bigcup_{j=1}^{\infty} R^j = R^1 \cup R^2 \cup \ldots \cup R^j \cup \ldots = \{(x, y) \mid (\exists p)((x, y) \in R^p)\}.
\]

You can prove by induction on $n$ that if $(x, y)$ gets into $tc(R)$ by $n$ or fewer rule applications (i.e., if $(x, y, n) \in \hat{tc}(R)$), then for some $p$, $(x, y) \in R^p$. Conversely, if $(x, z)$ is in $R^p$, then there is a proof using some finite number of steps showing that $(x, z) \in tc(R)$. 
Path interpretation of $R^n$

- This interpretation uses the directed graph representation of a relation $R$.
- A path in a digraph is a sequence $a_0, a_1, \ldots, a_p$ of nodes, such that each pair $(a_i, a_{i+1})$ in the sequence is an arrow in the graph from $a_i$ to $a_{i+1}$. The length of the path is the number $p$.
- We have that $R^n$ is the set of pairs $(a, b)$ such that there is a length $n$ path in the graph of $R$ from $a$ to $b$.
- So, the transitive closure of $R$ is the set of pairs $(a, b)$ such that $a$ is connected to $b$ by a path of some finite length.
- The Kevin Bacon number of an actor $a$ is the minimum $n$ such that $(a, kevinbacon) \in R^n$ where $R$ is the relation determined by the Internet Movie Data Base.
A formula for the transitive closure

- if $A$ has only $n$ elements then

$$tc(R) = \bigcup_{j=1}^{n} R^j.$$  

(This can be proved by induction, or, it’s obvious because if $a$ is connected to $b$, then you can get from $a$ to $b$ by a path that doesn’t repeat a node in the middle of it)

- Let $[R]$ be the matrix of $R$. Then the matrix of the transitive closure

$$[tc(R)] = \bigvee_{j=1}^{n} [R]^j.$$  

- We can turn this formula into an algorithm for computing the transitive closure.
The naive tc algorithm

• To compute $R^k$: first compute $R^1 = R$; then

$$
\bigcup_{i=1}^{k+1} R^i = R \cup R \circ \bigcup_{i=1}^{k} R^i
$$

• So with matrices, we let $M_k$ be the matrix of $\bigcup_{i=1}^{k} R^i$; thus

$$
M_{k+1} = M_1 \lor M_k M_1.
$$

• It takes $2n^3 - n$ operations to compute $M_k M_1$ and $n^2$ operations to add in $M_1$ for a total of $2n^3 + n^2 - n$ on each iteration, and there are $n - 1$ iterations, so we get $(n - 1)(2n^3 + n^2 - n) = 2n^4 - n^3 + n$ operations in all. This number is dominated by the $2n^4$ term.

• There’s a better algorithm, called Warshall’s algorithm, where we get a bound in which the first term is an $n^3$ term. Not that that’s so great!
Warshall’s Algorithm

- This method is grounded in the digraph representation of a relation \( R \).
- It saves us having to reconsider arrows in the graph over and over again, when adding arrows to the transitive closure.
- It so happens, though, that we can easily translate this idea to the matrix representation.
- The basic idea is to add arrows carefully in order.
- We enumerate the underlying set \( A = \{a_1, \ldots, a_n\} \).
- We use the concept of path and interior vertex of a path. After each iteration \( j \) of the algorithm, we have an arrow from \( a \) to \( b \) just in case there is a path from \( a \) to \( b \) all of whose interior vertices are in \( \{a_1, \ldots, a_j\} \).
A vertex is interior for a path if it’s not either endpoint of the path.
Translating into an algorithm

- First iteration: add all the arrows in $R$.

- On each successive iteration $j$, we find arrows between all pairs of points $(a, b)$ such that there is a path from $a$ to $b$ with all interior points in $\{a_1, \ldots, a_j\}$.

- If there is a path from $a$ to $b$, there certainly is a path that doesn’t repeat a vertex. So we look only at non-repeating paths.

- This can be found from looking at what we computed at iteration $j - 1$. That got us all pairs of points $(c, d)$ such that there was a path from $c$ to $d$ where the interior points are in $\{a_1, \ldots, a_{j-1}\}$.

- If there’s a path from $a$ to $b$ with all interior points in $\{a_1, \ldots, a_j\}$, then either there’s a path with all interior points in $\{a_1, \ldots, a_{j-1}\}$, which we found already, or there is a path from $a$ to $b$ that includes $a_j$ just once: $(a, \ldots, a_j, \ldots, b)$.

- The path from $a$ to $a_j$ we accounted for before, and also the path from $a_j$ to $b$.

- Let $R_j$ be the relation we compute at iteration $j$. Then

$$R_j = R_{j-1} \cup \{(a, b) \mid (a, a_j) \in R_{j-1} \land (a_j, b) \in R_{j-1}\}.$$  

- We compute $W_j$ at each step $j$, where $W_j$ is the matrix of $R_j$. There are a total of $n$ steps, and at each step we recompute an $n \times n$ matrix. To do this requires 2 operations per entry of the matrix, giving $2n^2$ operations per big iteration, for a total of $2n^3$ operations.
Calculating Kevin Bacon numbers

• The crucial equation on the last slide:

\[ R_j = R_{j-1} \cup \{(a, b) \mid (a, a_j) \in R_{j-1} \land (a_j, b) \in R_{j-1}\}. \]

• In matrix form, let \( W_j(i, k) \) be the \( i, k \) entry in \( W_j \). Then

\[ W_j(i, k) = W_{j-1}(i, k) \lor (W_{j-1}(i, j) \land W_{j-1}(j, k)). \]

• The complete algorithm for transitive closure:

\[
\begin{align*}
\text{initialize } W \text{ to the matrix of } R; \\
\text{for } j = 1 \text{ to } n \text{ do} \\
\quad \text{for } i, k = 1 \text{ to } n \text{ do} \\
\quad \quad W(i, k) := W(i, k) \lor (W(i, j) \land W(j, k))
\end{align*}
\]

• For Kevin Bacon numbers, let the entries of \( W \) be integers representing path lengths. Then the code for finding the length of the shortest path between any two vertices is

\[
\begin{align*}
\text{initialize } W \text{ to the matrix of } R; \\
\text{for } j = 1 \text{ to } n \text{ do} \\
\quad \text{for } i, k = 1 \text{ to } n \text{ do} \\
\quad \quad W(i, k) := \min(W(i, k), (W(i, j) + W(j, k)))
\end{align*}
\]