STOCHASTIC ANALYSIS OF CYCLIC SCHEDULES:
ALGORITHMS AND EXAMPLES

By

R.A. Bowman\(^1\) and J.A. Muckstadt\(^2\)

\(^1\)Union College, Schenectady, NY 12308.

\(^2\)This author's research was supported in part by grants from the IBM Corporation and by the National Science Foundation grant DDM-8819542.
ABSTRACT

Bowman [1] and Bowman and Muckstadt [2] describe an approach for analyzing a cyclic schedule by modeling it as a Markov chain. In this paper, we present more complete algorithms corresponding to this approach and apply them to a simple example. The objective is to provide a supplemental presentation of the relevant theory as well as to describe an approach for implementing the theory. We will use the same notation and definitions that are used by Bowman and Muckstadt [2].
Theorem 1 of reference [2] says that a finite schedule can be modeled as a Markov chain. The algorithm for generating the states consists of starting with the initial state, determining which states could be entered from that state (which tasks can be completed obeying the precedence constraints), determining which states can be entered from those states, etc., until the absorbing state is generated, which represents completion of the schedule. Formally, the algorithm is:

**Step 0:**  \( s^1 \leftarrow [s_i; s_i = 0, \forall i], \)
\( S \leftarrow \{s^1\}, \)
\( r \leftarrow 1, \)
\( c \leftarrow 1. \)

**Step 1:**  For \( j = 1 \) to \( m: \)

If \( s_j = 0 \) and \( j = 1: \)

\( N_j \leftarrow 1 \) (the number of next arc that can be completed on machine \( j), \)
\( s^E \leftarrow s^c + \epsilon_j^c \) (the state being entered).

Else If \( s_j = 0 \) and \( j \neq 1: \)

\( N_j \leftarrow L(j - 1) + 1, \)
\( s^E \leftarrow s^c + N_j \epsilon_j. \)

Else If \( s_j \neq 0: \)

\( N_j \leftarrow s_j + 1, \)
\( s^E \leftarrow s^c + \epsilon_j. \)

If \( s^c_{M(a)} \geq a \ \forall \ a \in PR(a^{-1}) \) and \( N_j \leq L(j), \)

Then:

If \( s^E \in S: \)

\( q(c, k) \leftarrow 1/\theta_{N_j} \) where \( s^k = s^E, \)
\( q(c, c) \leftarrow q(c, c) - 1/\theta_{N_j}. \)
Else If $s^E \notin S$:

$$r \leftarrow r + 1,$$

$$s^r \leftarrow s^E,$$

$$S \leftarrow S \cup \{s^r\},$$

$$q(c,r) \leftarrow 1/\theta_{N_j},$$

$$q(c,c) \leftarrow q(c,c) - 1/\theta_{N_j}.$$

Step 2: $c \leftarrow c + 1.$

If $c > r$, stop.

Else, go to step 1.

This algorithm generates the states so that they are numbered in increasing order of the number of arcs that have been completed. This is advantageous for computer implementation of much of the analysis to follow because transitions can only be made from a state that is one "level" above the state to which the transition is made. For example, when checking in step 2 to see if the state has already been added to the state space, one need only check the states in the level before the state the transition is from. The notation $q_{ij}$ will be used to refer to the $(i,j)^{th}$ element of the infinitesimal generator matrix and is equivalent to $q(s^i, s^j)$.

Example: For the example schedule presented in Figure 1 of reference [2], assume all task times are exponential with $\theta_j = 1$ for all arcs $j$. Then the arcs correspond exactly to the tasks and the states are generated in the following order by the algorithm:

$$s^1 = [0,0] \hspace{1cm} (nothing \ completed), \hspace{1cm} (Level \ 0),$$

$$s^2 = [1,0], \hspace{1cm} (Level \ 1),$$

$$s^3 = [0,3], \hspace{1cm} (Level \ 1),$$

$$s^4 = [2,0], \hspace{1cm} (Level \ 2),$$
\[ s^5 = [1,3], \quad \text{(Level 2)}, \]
\[ s^6 = [2,3], \quad \text{(Level 3)}, \]
\[ s^7 = [1,4], \quad \text{(Level 3)}, \]
\[ s^8 = [2,4] \quad \text{(schedule completed)}, \quad \text{(Level 4)}. \]

The infinitesimal generator matrix is as follows:

\[
Q = \begin{bmatrix}
-2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & -2 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}.
\]

An easy, and powerful, measure to compute is the probability that each state is reached. The general algorithm, and its application to the example problem are as follows:

**Probability of Reaching State \( j \) \( (P(s^j)) \)**

\[
P(s^1) = 1, \\
P(s^j) = \sum_{i<j} P(s^i) \left( \frac{q_{ij}}{\lambda_i} \right), \quad j = 2,3,\ldots,n, \\
\text{where } n \text{ is the number of states in the state space.}
\]

**Example:** Continuing our simple example, we have:

\[
P(s^1) = 1, \\
P(s^2) = \frac{1}{2} P(s^1) = \frac{1}{2}, \\
P(s^3) = \frac{1}{2} P(s^1) = \frac{1}{2},
\]

\[ P(s^4) = \frac{1}{2} P(s^2) = \frac{1}{4}, \]
\[ P(s^5) = \frac{1}{2} P(s^2) + P(s^3) = \frac{3}{4}, \]
\[ P(s^6) = P(s^4) + \frac{1}{2} P(s^5) = \frac{5}{8}, \]
\[ P(s^7) = \frac{1}{2} P(s^5) = \frac{3}{8}, \]
\[ P(s^8) = P(s^6) + P(s^7) = 1. \]

Notice that the sums of probabilities across each level is 1. This will always be the case because, by definition of level, each level must be visited exactly once.

The expected time from the reaching of a state (assuming it is reached) until the schedule is completed can be computed with the following algorithm:

**Expected Time From State \( j \) to Schedule Completion (\( EF(s^j) \))**

\[
EF(s^n) = 0, \text{ and } \\
EF(s^j) = \frac{1}{q_{jj}} + \sum_{i > j} \left( \frac{q_{ji}}{q_{jj}} \right) EF(s^i) \text{ for } j = n-1, n-2, \ldots, 2, 1.
\]

Example: Again, continuing our simple example:

\[
EF(s^8) = 0, \\
EF(s^7) = \frac{1}{2} EF(s^8) = 1, \\
EF(s^6) = \frac{1}{2} EF(s^8) = 1 \\
EF(s^5) = \frac{1}{2} + \frac{1}{2} EF(s^6) + \frac{1}{2} EF(s^7) = \frac{3}{2}, \\
EF(s^4) = \frac{1}{2} + EF(s^6) = 2, \\
EF(s^3) = \frac{1}{2} EF(s^5) = \frac{5}{2}, \\
EF(s^2) = \frac{1}{2} + \frac{1}{2} EF(s^4) + \frac{1}{2} EF(s^5) = \frac{9}{4}, \\
EF(s^1) = \frac{1}{2} + \frac{1}{2} EF(s^2) + \frac{1}{2} EF(s^3) = \frac{23}{8}.
\]
\( \text{EF}(s^1) \) is the expected time to complete the schedule \((E(T))\). Alternatively, this could be computed as follows:

\[
E(T) = \sum_{j \neq n} P(s^j) \frac{1}{q_{jj}}.
\]

This, of course, is just the expected time spent in each state summed across the states. For the example problem, we have:

\[
E(T) = \frac{1}{2} P(s^1) + \frac{1}{2} P(s^2) + P(s^3) + P(s^4) + \frac{1}{2} P(s^5) + P(s^6) + P(s^7) = \frac{23}{8} \approx 2.88.
\]

This alternative approach is useful if one wants to know the expected fraction of the time that something of particular interest occurs. For example, machine 2 is waiting for material from machine 1 only when the system is in state 3. The expected fraction of time this happens is \( P(s^3) \left( \frac{1}{q_{33}} / E(T) \right) = \left( \frac{1}{2} \right) \left( \frac{1}{2} \right) / \left( \frac{23}{8} \right) = \frac{4}{23} \).

The expected square of the time from state \( s^j \) until the schedule is completed can be computed as follows:

**Expected Square of Time from State \( s^j \) Until Schedule Completion (SF(\( s^j \)))**

\[
SF(s^n) = 0, \quad \text{and}
\]

\[
SF(s^j) = \sum_{j > i} \left( \frac{q_{ji}}{-q_{jj}} \right) \left[ SF(s^i) + \frac{2}{q_{jj}} + \left( \frac{2}{-q_{jj}} \right) \text{EF}(s^i) \right].
\]

For the example problem, we have:

\[
SF(s^8) = 0,
\]

\[
SF(s^7) = \left( \frac{1}{1} \right) \left[ SF(s^8) + \frac{2}{1} + \frac{2}{1} \text{EF}(s^8) \right] = 2,
\]

\[
SF(s^6) = \left( \frac{1}{1} \right) \left[ SF(s^8) + \frac{2}{1} + \frac{2}{1} \text{EF}(s^8) \right] = 2,
\]
\[
SF(s^5) = \left(\frac{1}{2}\right) \left[SF(s^6) + \frac{2}{4} + \frac{2}{2} \cdot EF(s^6)\right] + \left(\frac{1}{2}\right) \left[SF(s^7) + \frac{2}{4} + \frac{2}{2} \cdot EF(s^7)\right] = \frac{7}{2},
\]
\[
SF(s^4) = \left(\frac{1}{2}\right) \left[SF(s^6) + \frac{2}{4} + \frac{2}{1} \cdot EF(s^6)\right] = 6,
\]
\[
SF(s^3) = \left(\frac{1}{2}\right) \left[SF(s^5) + \frac{2}{4} + \frac{2}{4} \cdot EF(s^5)\right] = \frac{17}{2},
\]
\[
SF(s^2) = \left(\frac{1}{2}\right) \left[SF(s^4) + \frac{2}{4} + \frac{2}{2} \cdot EF(s^4)\right] + \left(\frac{1}{2}\right) \left[SF(s^5) + \frac{2}{4} + \frac{2}{4} \cdot EF(s^5)\right] = 7,
\]
\[
SF(s^1) = \left(\frac{1}{2}\right) \left[SF(s^2) + \frac{2}{4} + \frac{2}{2} \cdot EF(s^2)\right] + \left(\frac{1}{2}\right) \left[SF(s^3) + \frac{2}{4} + \frac{2}{2} \cdot EF(s^3)\right] = \frac{85}{8}.
\]

The variance of the time to complete the schedule is then given by:

\[
\text{Var}(T) = \text{E}(T^2) - \text{E}(T)^2 = \frac{85}{8} - \left(\frac{23}{8}\right)^2 = \frac{151}{64} \approx 2.36
\]
\[
\sigma_T = \sqrt{\text{Var}(T)} \approx 1.54.
\]

The expected value and variance of the time until a state is reached (given that it is reached) can also be computed and, as shown in [2], this information can be calculated for events rather than states. Letting \(ETE(E^k)\) and \(STE(E^k)\) be the expected time and expected square of the time until event \(k\), and \(H(E^k)\) be the set of states for which event \(K\) has already happened, we have:

\[
ETE(E^k) = \sum_{i<j} \sum_{s^i \in H(E^k)} P(s^i) \left(\frac{q_{ij}}{-q_{ii}}\right) \left[ET(s^i) + \frac{1}{-q_{ii}}\right],
\]
\[
STE(E^k) = \sum_{i<j} \sum_{s^i \in H(E^k)} P(s^i) \left(\frac{q_{ij}}{-q_{ii}}\right) \left[ST(s^i) + \frac{2}{-q_{ii}} + \left(\frac{2}{-q_{ii}}\right)ET(s^i)\right].
\]

Example: Let \(k\) be the event that operation 4 starts in our example schedule. Then

\[
H(E^k) = \{5,6,7,8\}
\]
\[
ETE(E^k) = P(s^2) \left(\frac{q_{25}}{-q_{22}}\right) \left[ET(s^2) + \frac{1}{-q_{22}}\right] + P(s^3) \left(\frac{q_{25}}{-q_{33}}\right) \left[ET(s^3) + \frac{1}{-q_{33}}\right] + P(s^4) \left(\frac{q_{46}}{-q_{44}}\right) \left[ET(s^4) + \frac{1}{-q_{44}}\right] = \frac{3}{2} = 1.5.
\]
\[ STE(E^k) = P(s^2) \left( \frac{q_{25}}{q_{22}} \right) \left[ ST(s^2) + \frac{2}{q_{22}} \left( -\frac{2}{q_{32}} \right) ET(s^2) \right] \\
+ P(s^3) \left( \frac{q_{35}}{q_{33}} \right) \left[ ST(s^3) + \frac{2}{q_{33}} \left( -\frac{2}{q_{43}} \right) ET(s^3) \right] \\
+ P(s^4) \left( \frac{q_{46}}{q_{44}} \right) \left[ ST(s^4) + \frac{2}{q_{44}} \left( -\frac{2}{q_{54}} \right) ET(s^4) \right] = \frac{7}{2} = 3.5. \]

The arc (task) criticalities can be calculated using Theorem 3 of reference [2]. Part 1 gives the end conditions, part 2 tells how to compute the criticality of arc \( a_i \) conditioned on state \( s_j \) being reached \((C(a_i|s_j))\) and part 3 tells how to convert the conditional criticalities to the unconditional ones \((C(a_i))\).

Example: For our example problem, part 1 of Theorem 3 tells us:

\[ C(a^4|s^7) = 1 \quad \text{and} \quad C(a^2) = P(s^7) = \frac{3}{8} \quad (C(a^4|s^7) = 0). \]
\[ C(a^4|s^6) = 1 \quad \text{and} \quad C(a^4) = P(s^6) = \frac{5}{8} \quad (C(a^2|s^6) = 0). \]

Then, starting with states 6 and 7 (level 3) we work back through the lower numbered states and levels using part 2 of Theorem 3, as follows:

\[ C(a^2|s^5) = \frac{1}{2} \quad C(a^2|s^7) = \frac{1}{2}, \]
\[ C(a^4|s^5) = \frac{1}{2} \quad C(a^4|s^6) = \frac{1}{2}, \]
\[ C(a^3|s^4) = C(a^4|s^6) = 1, \]
\[ C(a^1|s^3) = C(a^2|s^5) + C(a^4|s^5) = 1, \]
\[ C(a^4|s^3) = C(a^4|s^5) = \frac{1}{2}, \]
\[ C(a^2|s^2) = \frac{1}{2} \quad C(a^2|s^5) = \frac{1}{4}, \]
\[ C(a^3|s^2) = \frac{1}{2} \quad C(a^3|s^4) + \frac{1}{2} \quad C(a^4|s^5) = \frac{3}{4}. \]
\[ C(a^1|s^1) = \frac{1}{2} C(a^2|s^2) + \frac{1}{2} C(a^1|s^3) = \frac{5}{8}, \]
\[ C(a^3|s^1) = \frac{1}{2} C(a^3|s^2) = \frac{3}{8}. \]

Part 3 of Theorem 3 then gives us the remaining unconditional arc criticalities:

\[ C(a^1) = P(s^1)[\frac{1}{2} C(a^2|s^2)] + P(s^3)[C(a^2|s^5) + C(a^4|s^5)] = \frac{5}{8}. \]
\[ C(a^3) = P(s^2)[\frac{1}{2} C(a^4|s^5)] + P(s^4)[C(a^4|s^6)] = \frac{3}{8}. \]

For this particular example, the step corresponding to part 3 of the theorem was unnecessary since the criticalities of tasks 2 and 4 were computed in part 1 and the criticalities of tasks 1 and 3 were given as \( C(a^1|s^1) \) and \( C(a^3|s^1) \) in part 2. This will only be the case for the first and last task on each machine, however, and the others will be calculated using the unconditioning provided by part 3. In computer implementations, the probabilities (\( P(s^j) \)'s) should be computed first in a forward pass through the \( Q \) matrix and then the unconditional criticalities from part 3 can be computed concurrently with conditional criticalities with a backward pass through the \( Q \) matrix. This will enable conditional criticalities to be stored for only two levels (current and next highest) at a time.

Observe that the conditional criticalities for each state sum to 1 if the sum is taken only across arcs that could complete from the state. For example, \( C(a^4|s^3) \) is defined but it is not possible for arc \( a^4 \) to complete from state \( s^3 \) (since arc \( a^4 \) requires both arcs \( a^1 \) and \( a^3 \) to be done) so \( C(a^4|s^3) \) would not be included in the sum described for state \( s^3 \). That this will always be the case can be seen by considering the enabling of any arc that remains to be completed but can't be completed from that state. If we trace back to the arc that enabled this arc and continue to trace back its “enabling path” it is obvious that exactly one of the arcs that could be completed from the state is on the path. Thus, exactly one of these arcs is on every possible path from the state. This observation provides the scaling necessary for assessing cyclic task criticalities.
This completes the example computations for the finite schedule of Figure 1 [2]. We will see that these non-cyclic algorithms provide the building blocks for the algorithms for analyzing a cyclic schedule. These algorithms will be demonstrated on the cyclic schedule of Figure 3 of reference [2]. The first step is to generate the states and the infinitesimal generator matrix for the Markov chain of theorem 4 of reference [2]. This can be done using the following algorithm.

1. Select any feasible state as a starting point.
2. Identify all states that can be reached in one feasible transition from that state.
3. For each state generated in step 2, identify all states from which the state could be reached in one transition.
4. Repeat steps 2 and 3 until no new states are generated. This gives you all states in two “levels” of the Markov chain. One level (call it Level 1) contains the initial feasible state and all states generated in the repeats of step 3. The other level contains all the states generated in the repeats of step 2.
5. Starting with the states in Level 1, use the non-cyclic generation algorithm to generate the state space and $Q$ matrix. If $|t|$ is the number of arcs in the network, then there will be $|t|$ levels. Number them 1 to $|t|$ in the order that they are generated.

Example: For the cyclic schedule of Figure 3, again assume that all arcs have exponentially distributed lengths with unit means. The algorithm generates the states and $Q$ matrix as follows:

$$\text{Step 1: } s^1 = \left( \begin{array}{c} 2,4 \\ 0,0 \end{array} \right).$$

This state was arbitrarily selected and means that task 2 has been completed on machine 1 and task 4 has been completed on machine 2 in the same cycle.
Step 2: \[ s^* = \begin{pmatrix} 1,4 \\ 0,1 \end{pmatrix} \] (if task 1 is completed from state 1).

\[ s^{**} = \begin{pmatrix} 2,3 \\ 0,1 \end{pmatrix} \] (if task 3 is completed from state 1).

Step 3: \[ s^2 = \begin{pmatrix} 1,3 \\ 0,1 \end{pmatrix} \] (state \( s^{**} \) could be reached from \( s^2 \) if task 2 is completed).

There are no other states that \( s^* \) or \( s^{**} \) can be reached from.

Step 4: The only state that can be reached from \( s^2 \) is \( s^{**} \). Thus, the only states in level 1 are \( s^1 \) and \( s^2 \) and the only states in level 2 are \( s^* \) and \( s^{**} \).

Step 5: \[ s^1 = \begin{pmatrix} 2,4 \\ 0,0 \end{pmatrix}, \quad s^2 = \begin{pmatrix} 1,3 \\ 0,1 \end{pmatrix}, \] (Level 1),

\[ s^3 = \begin{pmatrix} 1,4 \\ 0,-1 \end{pmatrix}, \quad s^4 = \begin{pmatrix} 2,3 \\ 0,1 \end{pmatrix}, \] (Level 2),

\[ s^5 = \begin{pmatrix} 2,4 \\ 0,-1 \end{pmatrix}, \quad s^6 = \begin{pmatrix} 1,3 \\ 0,0 \end{pmatrix}, \] (Level 3),

\[ s^7 = \begin{pmatrix} 2,3 \\ 0,0 \end{pmatrix}, \quad s^8 = \begin{pmatrix} 1,4 \\ 0,0 \end{pmatrix}, \] (Level 4).

\[
Q = \begin{bmatrix}
-2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & -2 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & -2 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & -2 
\end{bmatrix}.
\]
Theorem 5 of reference [2], along with results from the non-cyclic analysis, allows the following algorithm to be used to calculate $E(T), E(T^2), \text{ and } P(s^i)$, the ergodic probability of visiting state $i$ in a cycle:

1. Let $i = 1$.

2. Start the non-cyclic algorithms for calculating $P(s^i), E(T(s^i)), \text{ and } ST(s^i)$ in the $i$th state of level 1.

3. Run these algorithms except stop after $|f|$ transitions return you to level 1 (there is no unique finishing state). This will calculate $P_{ij}$, (the probability the next visit to level 1 is in state $s^j$ given you start in state $s^i$), $E(T|i,j)$ (the expected time to return to level 1 given you start in state $s^i$ and end in state $s^j$) and $E(T^2|i,j)$ (the expected square of the same time) for all states $s^j \in \text{ level 1}$ as well as $P(j|i)$ (the probability of visiting state $s^j$ not in level 1 given you start in state $s^i$ in level 1) for all states $s^j$ not in level 1.

4. If $i < \text{(number of states in level 1)}, \text{ set } i = i+1 \text{ and go to step 2}.$

5. Solve $\pi = \pi P, \sum_{i \in \text{level 1}} \pi_i = 1$ for $\pi$.

6. Use $\pi$ and the results from step 3 to calculate $E(T), E(T^2), \text{ and } \pi_i^* \forall \text{ states } s^j \not\in \text{ level 1}$.

Note that the level chosen as level 1 is arbitrary. To make this algorithm as efficient as possible, one should choose the level with the fewest number of states as level 1.

Example: For the small cyclic example pictured in Figure 3, we can calculate the steady state probabilities of visiting each state in one cycle and the ergodic expected cycle time as follows:

$P(1|1) = 1,$

$P(2|1) = 0,$

$P(3|1) = \frac{1}{2} P(1|1) = \frac{1}{2},$

$P(4|1) = \frac{1}{2} P(1|1) = P(2|1) = \frac{1}{2},$
\[ P(5|1) = \frac{1}{2} \quad P(3|1) = \frac{1}{4}, \]
\[ P(6|1) = \frac{1}{2} \quad P(3|1) = P(4|1) = \frac{3}{4}, \]
\[ P(7|1) = P(5|1) = \frac{1}{2} \quad P(6|1) = \frac{5}{8}, \]
\[ P(8|1) = \frac{1}{2} \quad P(6|1) = \frac{3}{8}. \]

\[ P(1 \text{ in the next cycle}|1) = P(7|1) + \frac{1}{2} \quad P(8|1) = \frac{13}{16} = P_{11}. \]
\[ P(2 \text{ in the next cycle}|1) = \frac{1}{2} \quad P(8|1) = \frac{3}{16} = P_{12}. \]

\[ ET(1|1) = 0, \]
\[ ET(3|1) = \frac{-1}{q_{11}} + ET(1|1) = \frac{1}{2}, \]
\[ ET(4|1) = \frac{-1}{q_{11}} + ET(1|1) = \frac{1}{2}, \]
\[ ET(5|1) = \frac{-1}{q_{33}} + ET(3|1) = 1, \]
\[ ET(6|1) = \frac{1}{3} \left[ \frac{-1}{q_{33}} + ET(3|1) \right] + \frac{2}{3} \left[ \frac{-1}{q_{44}} + ET(4|1) \right] = \frac{4}{3}, \]
\[ ET(7|1) = \frac{2}{3} \left[ \frac{-1}{q_{55}} + ET(5|1) \right] + \frac{3}{3} \left[ \frac{-1}{q_{66}} + ET(6|1) \right] = \frac{19}{10}. \]
\[ ET(8|1) = \frac{-1}{q_{66}} + ET(6|1) = \frac{11}{6}. \]

\[ ET(1 \text{ in the next cycle}|1) = \frac{10}{13} \left[ \frac{-1}{q_{77}} + ET(7|1) \right] + \frac{3}{13} \left[ \frac{-1}{q_{88}} + ET(8|1) \right] \]
\[ = \frac{36}{13} = ET(T_{1,1}), \]
\[ ET(2 \text{ in the next cycle}|1) = \frac{-1}{q_{88}} + ET(8|1) \]
\[ = \frac{7}{3} = ET(T_{2,1}). \]

Similarly, if we started the non-cyclic algorithms in state 2, we would get:

\[ P(1|2) = 0, \quad ET(2|2) = 0, \]
\[ P(2|2) = 1, \quad ET(4|2) = 1, \]
\[ P(3|2) = 0, \quad ET(6|2) = 2, \]
\[ P(4|2) = 1, \quad ET(7|2) = \frac{5}{2}. \]
\begin{align*}
P(5|2) &= 0, & E(T|8|2) &= \frac{3}{2}, \\
P(6|2) &= 1, & E(T|2,1) &= \frac{10}{3}, \\
P(7|2) &= \frac{1}{2}, & E(T|2,2) &= 3, \\
P(8|2) &= \frac{1}{2}, & \\
P_{21} &= \frac{3}{4}, & \\
P_{22} &= \frac{1}{4}, & \\
\pi P = \pi \Rightarrow \pi = \left[ \frac{4}{9} \frac{1}{3} \right]. & \\
\Rightarrow \quad P(1) &= \frac{4}{9}, \\
P(2) &= \frac{1}{5}, \\
P(3) &= \frac{4}{9}\left(\frac{1}{2}\right) + \frac{1}{5}(0) = \frac{2}{9}, \\
P(4) &= \frac{4}{9}\left(\frac{1}{2}\right) + \frac{1}{5}(1) = \frac{3}{5}, \\
P(5) &= \frac{4}{9}\left(\frac{1}{4}\right) + \frac{1}{5}(0) = \frac{1}{5}, \\
P(6) &= \frac{4}{9}\left(\frac{3}{4}\right) + \frac{1}{5}(1) = \frac{4}{9}, \\
P(7) &= \frac{4}{9}\left(\frac{5}{8}\right) + \frac{1}{5}\left(\frac{1}{2}\right) = \frac{3}{5}, \\
P(8) &= \frac{4}{9}\left(\frac{3}{8}\right) + \frac{1}{5}\left(\frac{1}{2}\right) = \frac{2}{5}. \\
E(T) &= P(1)[P_{11}E(T|1,1) + P_{12}E(T|1,2)] \\
&\quad + P(2)[P_{21}E(T|2,1) + P_{22}E(T|1,2)] = \frac{14}{5}.
\end{align*}

Observe that the cycle time is \(\frac{14}{5}\) whereas the expected completion time for this network was \(\frac{23}{8}\) before it was converted to a cyclic network. It will always be the case that the expected time for the cyclic network is less than the corresponding one for the non-cyclic network as long as the cyclic schedule allows a subsequent cycle to be started before all operations in the previous cycle are completed. If not, the cyclic network is essentially repeats of the non-cyclic network so that the expected cycle time is the same as the expected completion time of the non-cyclic network.
Theorem 6 of reference [2] can then be used to calculate the cyclic task criticalities for the cyclic schedule, using the following algorithm:

1. Select a base level (either level 1 or the level with the fewest states for efficiency purposes).

2. Use the non-cyclic criticality algorithm to calculate $C(a^j, s^n | a^i, s^m)$ (the cyclic criticality of arc $a^j$ from state $s^n$ given that arc $a^i$ is critical from state $s^m$ in the next visit to the base level) for all states $s^n, s^m$ in the base level and all arcs that could be completed next for these states. Do this by successively setting $CC(a^i, s^m)$ (the cyclic criticality of arc $a^i$ from state $s^m$) = 1 and all other conditional criticalities in the base level equal to 0 and repeat for each $i, m$ combination. This requires one sweep through the non-cyclic algorithm for each $i, m$ combination. Each sweep gives you $C(a^j, s^n | a^i, s^m) \forall j, n$.

3. Use (1), (2) from the theorem to calculate $CC(a^j, s^n) \forall j, n$ in the base level.

4. Using $CC(a^j, s^n)$s as the end conditions, use the non-cyclic algorithm (which now applies (3), (4) from the theorem) to calculate $CC(a^j)$ (the cyclic criticality of arc $j$) $\forall j$.

Note that the conditional criticalities for one level serve the same purpose for the cyclic network as the conditional criticalities at the end of the network did for the finite network.

Example: Continuing with our cyclic example, Part 1 of Theorem 6 tells us (starting with level 4 and working back to level 1):

$$CC(a^2, s^8) = \frac{1}{2} CC(a^1, s^1) + \frac{1}{2} CC(a^2, s^2),$$

$$CC(a^3, s^8) = \frac{1}{2} CC(a^3, s^1),$$

$$CC(a^1, s^7) = CC(a^1, s^1),$$

$$CC(a^4, s^7) = CC(a^1, s^1) + CC(a^3, s^1),$$

$$CC(a^2, s^6) = \frac{1}{2} CC(a^2, s^8) + \frac{1}{4} CC(a^1, s^1) + \frac{1}{4} CC(a^2, s^2),$$

$$CC(a^4, s^6) = \frac{1}{2} CC(a^4, s^7) + \frac{1}{2} CC(a^3, s^8) + \frac{1}{2} CC(a^1, s^1) + \frac{3}{4} CC(a^3, s^1),$$

$$CC(a^1, s^5) = CC(a^1, s^7) = CC(a^1, s^1),$$
\[ CC(a^3, s^5) = CC(a^4, s^7) = CC(a^1, s^1) + CC(a^3, s^1), \]
\[ CC(a^1, s^4) = CC(a^2, s^6) + CC(a^4, s^6) + \frac{3}{4} CC(a^1, s^1) + \frac{1}{4} CC(a^2, s^2) + \frac{3}{4} CC(a^3, s^1), \]
\[ CC(a^4, s^4) = CC(a^1, s^1) + \frac{1}{2} CC(a^1, s^1) + \frac{3}{4} CC(a^3, s^1), \]
\[ CC(a^2, s^3) = \frac{1}{2} CC(a^2, s^6) + \frac{1}{8} CC(a^1, s^1) + \frac{1}{8} CC(a^2, s^2), \]
\[ CC(a^3, s^3) = \frac{1}{2} CC(a^3, s^5) + \frac{1}{2} CC(a^4, s^6) = \frac{3}{4} CC(a^1, s^1) + \frac{7}{8} CC(a^3, s^1), \]
\[ CC(a^2, s^2) = CC(a^1, s^4) = \frac{3}{4} CC(a^1, s^1) + \frac{1}{4} CC(a^2, s^2) + \frac{3}{4} CC(a^3, s^1), \]
\[ CC(a^4, s^2) = CC(a^1, s^4) = \frac{1}{2} CC(a^1, s^1) + \frac{3}{4} CC(a^3, s^1), \]
\[ CC(a^1, s^1) = \frac{1}{2} CC(a^2, s^3) + \frac{1}{2} CC(a^1, s^4) + \frac{7}{16} CC(a^1, s^1) + \frac{3}{16} CC(a^2, s^2) + \frac{3}{8} CC(a^3, s^1), \]
\[ CC(a^3, s^1) = \frac{1}{2} CC(a^3, s^3) = \frac{3}{8} CC(a^1, s^1) + \frac{7}{16} CC(a^3, s^1). \]

Notice that the last 4 equations involve only \( CC(a^2, s^2), CC(a^4, s^2), CC(a^1, s^1), \) and \( CC(a^3, s^1) \). The equations are not independent, however, and this is where part 2 of Theorem 6 comes in. Part 2 tells us:

\[ CC(a^1, s^1) + CC(a^3, s^1) = 1, \]
\[ CC(a^2, s^2) = 1. \]

Taking these two equations in combination with the last 4 equations from part 1, we can easily solve for the 4 unknowns and then use these values to solve for all the conditional criticalities. The solution is:

\[ CC(a^1, s^1) = \frac{3}{5}, \quad CC(a^1, s^4) = 1, \quad CC(a^2, s^6) = \frac{2}{5}, \quad CC(a^2, s^8) = \frac{4}{5}, \]
\[ CC(a^3, s^1) = \frac{2}{5}, \quad CC(a^4, s^4) = \frac{3}{5}, \quad CC(a^4, s^6) = \frac{3}{5}, \quad CC(a^3, s^8) = \frac{1}{5}, \]
\[ CC(a^2, s^2) = 1, \quad CC(a^2, s^3) = \frac{1}{5}, \quad CC(a^1, s^5) = \frac{3}{5}, \quad CC(a^1, s^7) = \frac{3}{5}, \]
\[ CC(a^4, s^2) = \frac{3}{5}, \quad CC(a^3, s^3) = \frac{4}{5}, \quad CC(a^3, s^5) = 1, \quad CC(a^4, s^7) = 1. \]

Now we can use Part 4 of Theorem 6 to solve for the unconditional cyclic criticalities, as follows:
CC(a^{1}) = P(s^{4})[CC(a^{2},s^{6}) + CC(a^{4},s^{6})] + P(s^{1})[\frac{1}{2}CC(a^{2},s^{3})] = \frac{17}{25},

CC(a^{2}) = P(s^{8})[\frac{1}{2}CC(a^{1},s^{1}) + P(s^{2})][CC(a^{1},s^{4})] = \frac{8}{25},

CC(a^{3}) = P(s^{5})[CC(a^{4},s^{7}) + P(s^{3})][CC(a^{4},s^{6})] = \frac{8}{25},

CC(a^{4}) = P(s^{7})[CC(a^{1},s^{1}) + CC(a^{3},s^{1})] + P(s^{6})[\frac{1}{2}CC(a^{3},s^{8})] = \frac{17}{25}.

Bowman and Muckstadt [2] describe how to use the task criticalities, expected value and variance of the schedule completion (cycle) time, and other information to improve the schedule performance. They also report computational experience from a computer implementation of the algorithms that were applied to small examples in this paper.

REFERENCES
