COMP9334
Capacity Planning for Computer Systems and Networks

Week 3B Compute the
state balance equati matically

Deriving state balance equations

- The aim of this document is to explain how you can derive the following equations of the database server automatically

$$6\,P_{(2,0,0)} - 4\,P_{(1,1,0)} - 2\,P_{(1,0,1)} + 0\,P_{(0,2,0)} + 0\,P_{(0,1,1)} + 0\,P_{(0,0,2)} = 0$$

$$-3\,P_{(2,0,0)} + 10\,P_{(1,1,0)} + 0\,P_{(1,0,1)} - 4\,P_{(0,2,0)} - 2\,P_{(0,1,1)} + 0\,P_{(0,0,2)} = 0$$

$$-3\,P_{(2,0,0)} + 0\,P_{(1,1,0)} \quad P_{(0,1,1)} - 2\,P_{(0,0,2)} = 0$$

$$0\,P_{(2,0,0)} - 3\,P_{(1,1,0)} + 0\,P_{(1,0,1)} + 4\,P \quad {}_{(0,1,1)} + 0\,P_{(0,0,2)} = 0$$

$$0\,P_{(2,0,0)} - 3\,P_{(1,1,0)} - 3\,P_{(1,0,1)} + 0\,P_{(0,2,0)} + 6\,P_{(0,1,1)} + 0\,P_{(0,0,2)} = 0$$

$$0\,P_{(2,0,0)} + 0\,P_{(1,1,0)} - 3\,P_{(1,0,1)} + 0\,P_{(0,2,0)} + 0\,P_{(0,1,1)} + 2\,P_{(0,0,2)} = 0$$

Deriving state balance equations

- We rewrite the equations from the last page as a matrix

$$
\begin{array}{c|cccccc}
 & (2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2) \\
\hline
(2,0,0) & 6 & -4 & -2 & 0 & 0 & 0 \\
(1,1,0) & -3 & 10 & 0 & -4 & -2 & 0 \\
(1,0,1) & -3 & 0 & 8 & 0 & -4 & -2 \\
(0,2,0) & 0 & & 0 & 0 & & \\
(0,1,1) & 0 & & 6 & 0 & & \\
(0,0,2) & 0 & 0 & -3 & & 0 & 2
\end{array}
$$

- This matrix is sufficient for you to solve the equations, so this matrix is what we need
- We have also added row labels (in red) and column labels (in green)
- Let us first understand the meaning of the row and column labels

Meaning of the row labels

- The row label (2,0,0) means the corresponding row comes from the state balance equation for the state (2,0,0)
- The meaning is similar for the other rows

$$
\begin{array}{c}
(2,0,0) \\ (1,1,0) \\ (1,0,1) \\ (0,2,0) \\ (0,1,1) \\ (0,0,2) \\ \\
\end{array}
\begin{array}{cccccc}
\multicolumn{6}{c}{\left[\begin{array}{cccccc}
6 & -4 & -2 & 0 & 0 & 0 \\
-3 & & & & -2 & 0 \\
-3 & & & & -4 & -2 \\
0 & -3 & 0 & & 0 & 0 \\
0 & -3 & -3 & & 6 & 0 \\
0 & 0 & -3 & 0 & 0 & 2
\end{array}\right]} \\
(2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2)
\end{array}
$$

Meaning of the column labels

- The column label (2,0,0) means the numbers in the column should be multiplied by $P_{(2,0,0)}$
- Similarly for the other columns
- You can check that by going back to Page 1

$$
\begin{array}{c c}
(2,0,0) & \left[\begin{array}{cccccc} 6 & -4 & -2 & 0 & 0 & 0 \\ -3 & & & & -2 & 0 \\ -3 & & & & -4 & -2 \\ 0 & -3 & 0 & & 0 & 0 \\ 0 & -3 & -3 & & 6 & 0 \\ 0 & 0 & -3 & 0 & 0 & 2 \end{array}\right] \\
(1,1,0) & \\
(1,0,1) & \\
(0,2,0) & \\
(0,1,1) & \\
(0,0,2) & \\
 & \begin{array}{cccccc} (2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2) \end{array}
\end{array}
$$

Signs of the number

- Note that
  - All diagonal elements have positive values
  - All non-diagonal elements have negative or zero values

	(2,0,0)	(1,1,0)	(1,0,1)	(0,2,0)	(0,1,1)	(0,0,2)
(2,0,0)	[6	-4	-2	0	0	0
(1,1,0)	-3				-2	0
(1,0,1)	-3				-4	-2
(0,2,0)	0	-3	0		0	0
(0,1,1)	0	-3	-3		6	0
(0,0,2)	0	0	-3	0	0	2]

Let us now examine where the non-diagonal elements are coming from

The number 4 is the transition rate from state (1,1,0) (column label) to state (2,0,0) (row label)

The number 2 is the transition rate from state (1,0,1) (column label) to state (2,0,0) (row label)

	(2,0,0)	(1,1,0)	(1,0,1)	(0,2,0)	(0,1,1)	(0,0,2)
(2,0,0)	[6				0	0
(1,1,0)	-3	10	0		-2	0
(1,0,1)	-3	0	8		-4	-2
(0,2,0)	0	-3	0	4	0	0
(0,1,1)	0	-3	-3	0	6	0
(0,0,2)	0	0	-3	0	0	2]

Using nested for-loops to find the non-diagonal elements

- On the last slide, we say that the off-diagonal elements is given by (-1) * (transition rate of the column label to the row label)
- This suggests that we can use a nested for-loop to obtain the off-diagonal elements. The pseudo-code is:


```
for i from 1 to 6          # r
    for j from 1 to 6      #
        if i not equal j    # Off-diagonal ele
            col_i = label of i-th row
            row_j = label of j-th row
            determine the transition rate from col_i to col_j
```


- The next question is how you can determine the transition rate by using the row and column labels.

In order to understand how you can determine the transition rate given the row and column labels, let us focus on the transitions that give the number -4 in the matrix. The table below summarises all the three possible transitions. We find that if we do an elementwise subtraction of the row label from the column label, we always get (1,-1,0). What does (1,-1,0) represent? Recall that the state is (#users in CPU,#users in fast disk,#users in slow disk). Therefore, the difference (1,-1,0) means a user has left the fast disk to go to the CPU. An important lesson is that we can use the column label minus the row label to tell us what the state transition should be.

Row label	Column label	Column label minus row label
(2,0,0)	(1,1,0)	(2,0,0) - (1,1,0) = (1,-1,0)
(1,1,0)	(0,2,0)	(1,1,0) - (0,2,0) = (1,-1,0)
(1,0,1)		0,1,1) = (1,-1,0)

$$
\begin{array}{c}
(2,0,0) \\ (1,1,0) \\ (1,0,1) \\ (0,2,0) \\ (0,1,1) \\ (0,0,2) \\ \\
\end{array}
\begin{array}{c}
\left[\begin{array}{cccccc}
6 & -4 & -2 & & 0 & 0 \\
-3 & 10 & 0 & -4 & -2 & 0 \\
-3 & 0 & 8 & 0 & -4 & -2 \\
0 & -3 & 0 & 4 & 0 & 0 \\
0 & -3 & -3 & 0 & 6 & 0 \\
0 & 0 & -3 & 0 & 0 & 2
\end{array}\right] \\
\begin{array}{cccccc}
(2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2)
\end{array}
\end{array}
$$

For the database server, there are four state transitions that have a non-zero transition rate. See the table below. All other transitions will have zero transition rate. For example, the transition from column label (0,2,0) to row label (1,0,1) has difference (1,-2,1). This difference is not listed in the first column below, so it has zero transition rate. The next slide shows the pseudo code that you use to find all the off-diagonal elements.

Column label minus row label	Meaning
(1,-1,0)	User finishes at fast disk and goes to CPU
(1,0,-1)	User finishes at slow disk and goes to CPU
(-1,1,0)	and goes to fast disk
(-1,0,1)	and goes to slow disk

$$
\begin{array}{c|c}
(2,0,0) & \\
(1,1,0) & \\
(1,0,1) & \\
(0,2,0) & \\
(0,1,1) & \\
(0,0,2) & \\
 &
\end{array}
\begin{array}{c}
\left[\begin{array}{cccccc}
6 & -4 & -2 & [illegible] & 0 & 0 \\
-3 & 10 & 0 & -4 & -2 & 0 \\
-3 & 0 & 8 & 0 & -4 & -2 \\
0 & -3 & 0 & 4 & 0 & 0 \\
0 & -3 & -3 & 0 & 6 & 0 \\
0 & 0 & -3 & 0 & 0 & 2
\end{array}\right] \\
\begin{array}{cccccc}
(2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2)
\end{array}
\end{array}
$$

Pseudo code to find the non-diagonal elements

```
for i from 1 to 6          # row index
    for j from 1 to 6      # column index
        if i not equal j    # Off-diagonal element
            col_i = label of i-th row
            row_j = label of j-th row
            transition = col_i – row_j
            if transition == (1,-1,0)
                # user move from
            else if transition == (
                # user move from slow disk to CPU
            else if transition == (-1,1,0)
                # user move from CPU to fast disk
            else if transition == (-1,0,1)
                # user move from CPU to slow disk
            else
                # transition rate is zero
```


Once you can determine the type of transition, you can fill in the transition rate which is a constant for a given transition. We still need to determine the diagonal elements.

The boxes below explains how the diagonal elements are coming from. This means that once we have identified a transition from the column label to a row label, we can accumulate it in the diagonal element corresponding to the column label.

You can see the complete code in rate_matrix_automated.py

The number 6 is the sum of
- transition rate from (2,0,0) to (1,1,0), and
- transition rate from (2,0,0) to (

The number 10 is the sum of
- transition rate from (1,1,0) to (2,0,0), and
- ate from (1,1,0) to (0,2,0), and
- ate from (1,1,0) to (0,1,1)

$$
\begin{array}{c|cccccc}
(2,0,0) & 6 & -4 & -2 & [illegible] & 0 & 0 \\
(1,1,0) & -3 & 10 & 0 & -4 & -2 & 0 \\
(1,0,1) & -3 & 0 & 8 & 0 & -4 & -2 \\
(0,2,0) & 0 & -3 & 0 & 4 & 0 & 0 \\
(0,1,1) & 0 & -3 & -3 & 0 & 6 & 0 \\
(0,0,2) & 0 & 0 & -3 & 0 & 0 & 2 \\
\hline
 & (2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2)
\end{array}
$$

You can see the complete code in rate_matrix_automated.py

Finally, you may have noticed already that the sum of each column is 0. This is a consequence of state balance. This implies the following sanity check: If the column sum of the matrix that you have obtained is non-zero, then something is wrong.

$$
\begin{array}{c|cccccc}
(2,0,0) & 6 & -4 & -2 & & 0 & 0 \\
(1,1,0) & -3 & 10 & 0 & -4 & -2 & 0 \\
(1,0,1) & -3 & 0 & 8 & 0 & -4 & -2 \\
(0,2,0) & 0 & -3 & 0 & 4 & 0 & 0 \\
(0,1,1) & 0 & -3 & -3 & 0 & 6 & 0 \\
(0,0,2) & 0 & 0 & -3 & 0 & 0 & 2 \\
\hline
 & (2,0,0) & (1,1,0) & (1,0,1) & (0,2,0) & (0,1,1) & (0,0,2)
\end{array}
$$