Problem 1

(a) F (there is non-negligible variance for the number of keys per node)
(b) T
(c) T
(d) F (it is still possible when appropriately selecting the partitions of servers)
(e) F
(f) F (HDFS is the reason why)
(g) F (Spanner provides stronger transactional guarantees than BigTable)
(h) T
(i) T
(j) T
Problem 2

(a) 0.85

Let \( x \) be the event that node \( x \) is fully operational; \( \bar{x} \) be the event that node \( x \) is not available for the full period. Node \( d \) is always up, so we omit it in the following discussion. A write transaction can complete successfully if one of the following events occurs:

\[
abc \\
\bar{a}bc \\
ab\bar{c} \\
ab\bar{c}
\]

The probabilities of these events are

\[
0.5 \times 0.8 \times 0.9 = 0.36 \\
0.5 \times 0.8 \times 0.9 = 0.36 \\
0.5 \times 0.2 \times 0.9 = 0.09 \\
0.5 \times 0.8 \times 0.1 = 0.04
\]

respectively.
Problem 2

(b) Yes.
A sample vote assignment: a – 2 votes, b – 2 votes, c – 1 vote, d – 1 vote.

Note: This question has more than one valid vote assignments.

(c) {{d}}. 1.
Problem 3

(a) the optimal allocation is sending 15 nodes on A and 15 on B, while cutting only 3 edges. One way is to send one connected component to A and one to B, while splitting the third one into 5 and 5 nodes by cutting 3 edges.

Communication Cost: 3  Imbalance Cost: 10

(b) the optimal allocation is splitting in half, i.e., 5 nodes on A and 5 nodes on B.
Problem 4

(a)

<table>
<thead>
<tr>
<th></th>
<th>A2</th>
<th>B1</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>{(5,11)}</td>
<td>{(7,5)}</td>
</tr>
<tr>
<td>7</td>
<td>{(7,14), (7,12)}</td>
<td>{(7,7), (7,7)}</td>
</tr>
</tbody>
</table>

(b) FILTER G1 BY (SUM(A2.X) == SUM(B1.Y))
Problem 5

(a) 35
(b) 4 or 5
(c) 6

Note: There is no need to solve (b) and (c) analytically. The most straightforward way to solve (b) and (c) is to observe that the cost is a convex function (via the 2nd derivative or by just understanding the tradeoff in the cost function), and then examine the small range of integer values near the optimum.
Problem 6(a)

<table>
<thead>
<tr>
<th></th>
<th>1-version</th>
<th>2-version</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>S2</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>S3</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>S4</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>S5</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>S6</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
Problem 6(b)

<table>
<thead>
<tr>
<th></th>
<th>Simple TO</th>
<th>Strict TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>S12</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>S13</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>S14</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
Problem 7

(a) like.

Map(t)
    emit(t.photo_ID, 1)

Reduce(key, val)
    emit(key, count(val))
Problem 7

(b)
Map(t)
  emit(t.count, t.photo-ID)

Reduce(key, val)
  emit(key, val)
Problem 7

(c) like.

Map(t)
    emit(t.fan_ID, t.photo_ID)

Reduce(key, val)
    if(val.size == 1 && the first element in val is ‘best’)
        emit(key)