## Gaussian elimination in binary arithmetic

**Problem solution:**
Time can differ based on your implementation.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Result</th>
<th>Elapsed time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>No solution.</td>
<td>0.001979 seconds</td>
</tr>
<tr>
<td>A2</td>
<td>More than one solution.</td>
<td>0.015866 seconds</td>
</tr>
<tr>
<td>A3</td>
<td>One solution.</td>
<td>0.013927 seconds</td>
</tr>
<tr>
<td>A4</td>
<td>No solution.</td>
<td>0.013712 seconds</td>
</tr>
<tr>
<td>A5</td>
<td>More than one solution.</td>
<td>0.013975 seconds</td>
</tr>
<tr>
<td>A6</td>
<td>One solution.</td>
<td>0.014115 seconds</td>
</tr>
<tr>
<td>A7</td>
<td>No solution.</td>
<td>0.014189 seconds</td>
</tr>
<tr>
<td>A8</td>
<td>No solution.</td>
<td>0.026557 seconds</td>
</tr>
<tr>
<td>A9</td>
<td>One solution.</td>
<td>0.056685 seconds</td>
</tr>
<tr>
<td>A10</td>
<td>One solution.</td>
<td>0.338309 seconds</td>
</tr>
<tr>
<td>A11</td>
<td>More than one solution.</td>
<td>0.438238 seconds</td>
</tr>
</tbody>
</table>
Reference implementation:

```matlab
function x = bin_gauss_solve(A, b)
% solver for system of linear equations in binary arithmetic using Gaussian
% elimination algorithm.

mat = [A b];
[m n] = size(A); % read the size of the original matrix A

for i = 1 : n
    j = find(mat(i:m, i), 1); % finds the FIRST 1 in i-th column starting at i
    if isempty(j)
        error('More than one solution.');
    else
        j = j + i - 1; % we need to add i-1 since j starts at i
        temp = mat(j, :); % swap rows
        mat(j, :) = mat(i, :);
        mat(i, :) = temp;
        % add i-th row to all rows that contain 1 in i-th column
        % starting at j+1 - remember up to j are zeros
        for k = find(mat( (j+1):m, i ))'
            mat(j + k, :) = bitxor(mat(j + k, :), mat(i, :));
        end
    end
end

% remove all-zero rows if there are some
mat = mat( sum(mat,2)>0 ,:);

if any(sum(mat(:,1:n), 2)==0) % no solution because matrix A contains
    error('No solution.'); % all-zero row, but with nonzero RHS
end

% calculate final solution
x = zeros(n, 1); % just an initialization
for i = n : -1 : 1 % go back from n to 1
    x(i) = bitand(dot(mat(i, i:n), x(i:n)) + mat(i, n + 1), 1);
end
end
```