function fuzzy\_sequencing()

% Define fuzzy sets (rows represent jobs and columns represent machines)

% Fuzzy times for 5 jobs on 3 machines M1, M2, M3

M1 = {[-2,-1,0,1,4,5,6,7], [1,2,3,4,5,6,9,10], [1,2,4,5,8,10,12,14], [2,3,6,7,8,10,11,13], [3,5,6,8,10,12,13,15]};

M2 = {[-3,-2,-1,0,1,2,3,4], [-3,-2,-1,0,4,5,6,7], [-4,-3,0,1,2,3,4,5], [-2,-1,0,1,2,3,4,5], [-1,0,1,2,3,4,5,6]};

M3 = {[7,8,10,11,14,15,17,18], [6,8,9,10,12,13,14,16], [-1,0,1,2,4,5,6,7], [5,6,7,8,12,13,14,15], [1,2,3,4,5,6,7,8]};

% Define alpha level (0 to 1)

alpha = 0.1; % You can change this value

% Calculate the robust ranking for each fuzzy time

ranked\_M1 = cellfun(@(fuzzy\_set) robust\_ranking(fuzzy\_set, alpha), M1);

ranked\_M2 = cellfun(@(fuzzy\_set) robust\_ranking(fuzzy\_set, alpha), M2);

ranked\_M3 = cellfun(@(fuzzy\_set) robust\_ranking(fuzzy\_set, alpha), M3);

% Display the rankings for each process

disp('Rankings for Cutting (M1):');

disp(ranked\_M1);

disp('Rankings for Sewing (M2):');

disp(ranked\_M2);

disp('Rankings for Pressing (M3):');

disp(ranked\_M3);

% Step 1: Create artificial machines A and B (for Johnson's rule)

Machine\_A = ranked\_M1 + ranked\_M2; % M1 + M2

Machine\_B = ranked\_M2 + ranked\_M3; % M2 + M3

% Step 2: Apply Johnson's rule for job sequencing

num\_jobs = length(ranked\_M1);

job\_sequence = zeros(1, num\_jobs); % To store the final job sequence

scheduled\_jobs = false(1, num\_jobs); % To keep track of scheduled jobs

front\_pointer = 1; % Pointer for front of the sequence

back\_pointer = num\_jobs; % Pointer for back of the sequence

for i = 1:num\_jobs

% Find unscheduled jobs and their processing times

unscheduled\_A = Machine\_A(~scheduled\_jobs);

unscheduled\_B = Machine\_B(~scheduled\_jobs);

% Find the minimum processing times in Machine A and B

[min\_A, idx\_A] = min(unscheduled\_A);

[min\_B, idx\_B] = min(unscheduled\_B);

unscheduled\_job\_indices = find(~scheduled\_jobs);

if min\_A <= min\_B

job\_index = unscheduled\_job\_indices(idx\_A); % Minimum on Machine A

job\_sequence(front\_pointer) = job\_index; % Schedule at front

front\_pointer = front\_pointer + 1; % Move front pointer

else

job\_index = unscheduled\_job\_indices(idx\_B); % Minimum on Machine B

job\_sequence(back\_pointer) = job\_index; % Schedule at back

back\_pointer = back\_pointer - 1; % Move back pointer

end

% Mark the job as scheduled

scheduled\_jobs(job\_index) = true;

end

% Display the optimal job sequence

fprintf('Optimal Job Sequence: ');

disp(job\_sequence);

end

% Robust Ranking Function for Fuzzy Sets

function rank = robust\_ranking(fuzzy\_set, alpha)

% Calculate alpha-cut interval

lower\_alpha = (1 - alpha) \* min(fuzzy\_set) + alpha \* max(fuzzy\_set);

upper\_alpha = alpha \* min(fuzzy\_set) + (1 - alpha) \* max(fuzzy\_set);

% Robust ranking index (centroid)

rank = (lower\_alpha + upper\_alpha) / 2;

end