Arena of Ratings: A Binary Search Tree Based Real-Time Matchmaking Engine

Author: Ishtiaq Ahmed
Instructor: Mr.M.Shayan Shah
Affiliation: Department of Artificial Intelligence, Institute of Management Sciences (IMSciences), Peshawar, Pakistan

PROBLEM STATEMENT

Design a real-time matchmaking engine for a competitive game "Arena of Ratings" that manages player data using a Binary Search Tree. The system must support dynamic operations like player join/leave, health updates, finding nearest opponents, range queries, ranking operations, and structural diagnostics - all ordered by unique player ratings.

SOLUTION

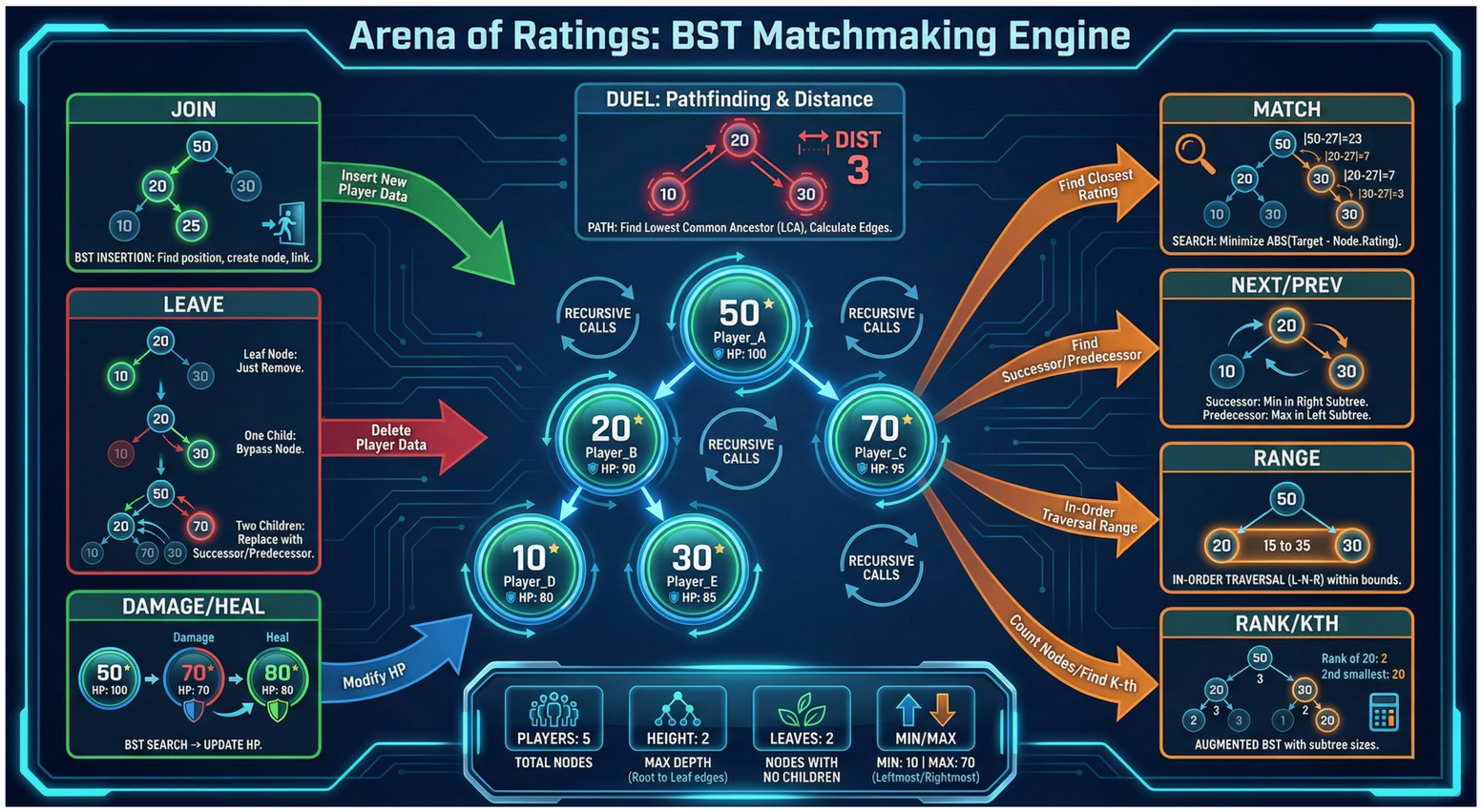
Implemented a custom Binary Search Tree (BST) data structure in C++ with pointer-based nodes. Each node stores player information (rating, name, HP) with left and right child pointers. The BST maintains sorted order by rating, enabling efficient O(log n) average-case operations for insertion, deletion, search, and traversal.

CONCLUSION

Successfully implemented a fully functional matchmaking engine using a custom Binary Search Tree that efficiently handles all required operations without using STL containers. The solution demonstrates proper BST fundamentals including recursive traversal, all three deletion cases, and advanced tree algorithms for ranking and path-finding. The system achieves optimal performance for game matchmaking scenarios, maintaining sorted player data and enabling fast queries essential for real-time competitive gaming environments.

WORKING METHODOLOGY

Data Structure: BST with nodes containing rating (key), name, and HP
Core Operations: Recursive insertion with duplicate checking, deletion handling all 3 cases (0, 1, 2 children), in-order traversal for sorted queries
Advanced Features: Predecessor/successor finding, rank calculation by counting nodes, k-th element via in-order traversal, path-finding for duel distance using LCA algorithm
Complexity: Most operations achieve O(log n) on balanced trees, O(n) worst case on skewed trees



KEY FEATURES

- **JOIN**: Insert players with duplicate prevention
- **LEAVE**: Delete with proper BST restructuring (in-order successor for 2-child case)
- **MATCH**: Find closest opponent by comparing predecessor and successor
- **RANGE**: In-order traversal for sorted output within bounds
- **RANK/KTH**: Counting and indexed access operations
- **DUEL**: Calculate tree path distance using common ancestor finding