# Common Search Strategies and Heuristics With Respect to the N-Queens Problem

by

Sheldon Dealy

# Topics

- Problem
- History of N-Queens
- Searches Used
- Heuristics
- Implementation Methods
- Results
- Discussion

# Problem

- Given an NxN chess board, can we place N Queens on the board such that no queen threatens another?
  - Yes, for N != 2,3
- Can it be made fast?
  - Problematic, search space is N^N.
- N-Queens is often implemented as a depth first search.

# Common Implementation Strategies

- Generate all solutions for a given N.
- Generate only the fundamental solutions.
  - Many solutions are isomorphic through rotation and reflection.
- Generate one or more, but not all solutions.

# Variations of N-Queens

- Some Problem Variations
  - Fewest number of queens to cover all squares.
  - Toroidal N-Queens (wrap around the board).
  - 3-D N-Queens (NxNxN board).
- Search Variations
  - Gradient based heuristics [5].
  - Heuristic repair approach implemented by Minton [7]. More on this later.

# History of N-Queens

- First reference to N-Queens problem was published in a German chess magazine by Max Bezzel, a chess player, in 1848.
- Gauss took a passing interest in the problem after reading an 1850 article written by Franz Nauck, who discovered all 92 solutions to the 8-Queens problem.
- Captured the interests of many others, including the mathematician J.W.L. Glaisher.

# Historical Sideline

- In an 1874 paper, J.W.L. Glaisher gave credit of the original N-Queens problem to Nauck, even though he had access to the correct facts [2].
- This erroneous historical account persists to this day, most recently in 2002[3].
- Moral: Your research is only as good as your references.

# Search Strategies

- Depth First Search (control)
- Depth First With Heuristics
  - Expand child nodes and backtrack like DFS except child nodes on stack in priority order.
- Beam Search
  - Beam width as a function of N.
- Branch and Bound
  - Upper bound is a limiting measure of worst acceptable proximity to a solution.
  - But in N-Queens, incremental improvement does not always suggest a solution.

# Heuristics

- H0. No heuristic
- H1. Distance from previously placed queen (local density)
  - Quick
  - Not very informed (stupid)
- H2. Mean distance between previously placed queens
  - Mean aggregation of H1
  - Still pretty fast
  - Still not very informed

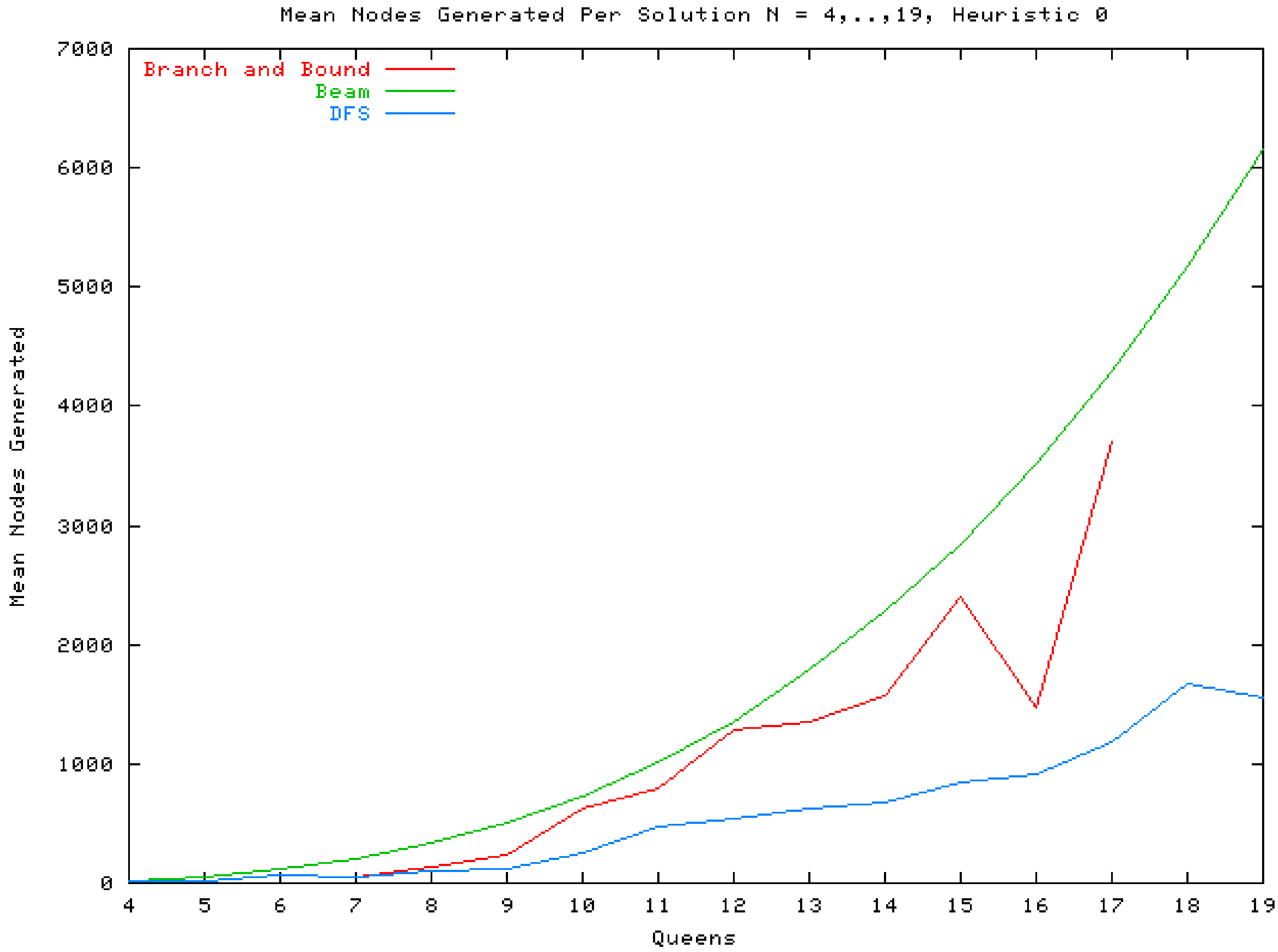
# Heuristics continued

- H3. Number of open squares on board
  - Inefficient to calculate or estimate
  - More informed
  - More open squares are better
- H4. Mean hamming distance between all queens on the board
  - Painful to calculate
  - More informed

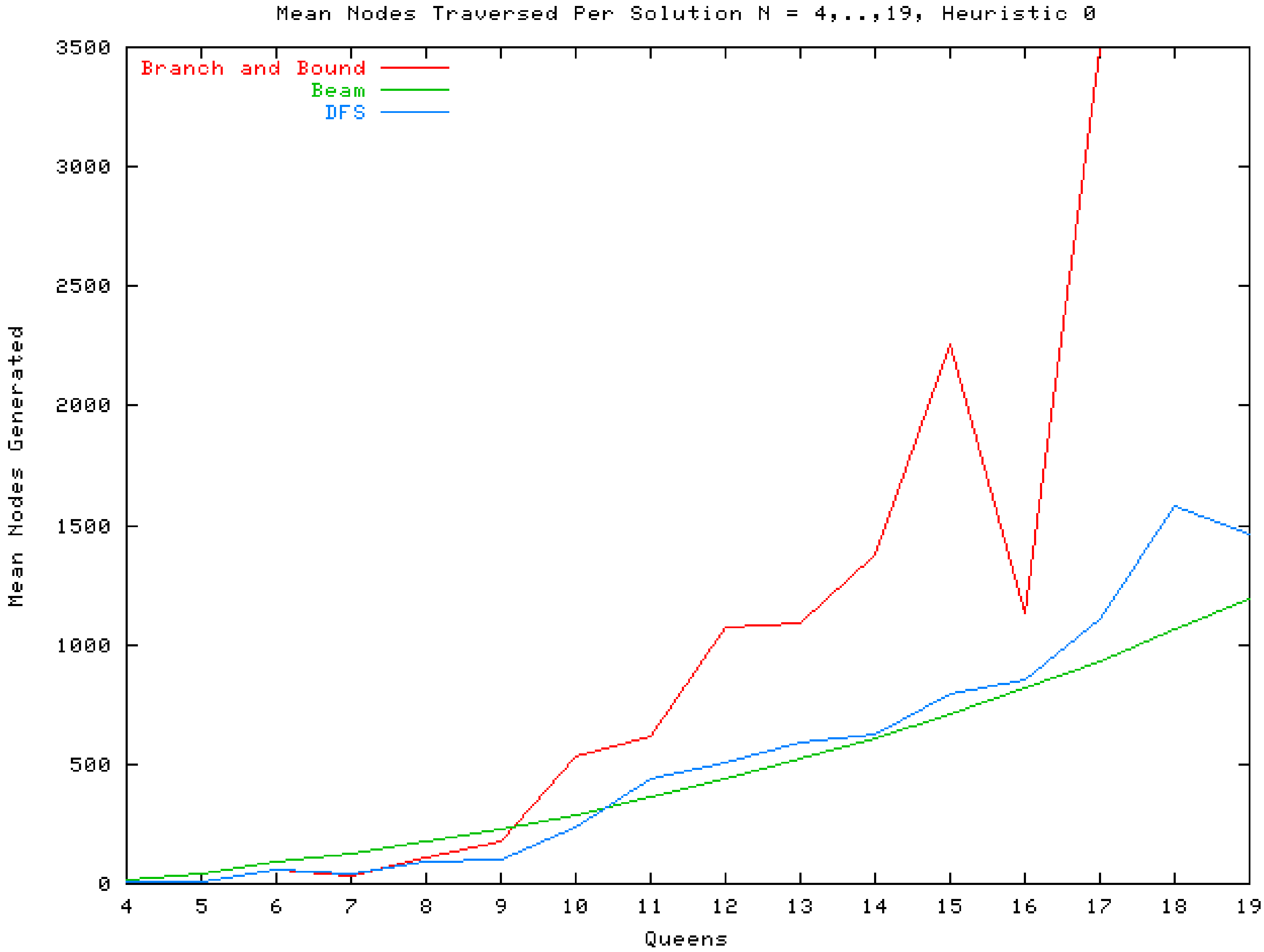
# Methods

- All searches use the same base code.
- Searches use different data structures to allow for different search traversals.
- Implementation allows for heuristics and searches to be mixed and matched.
- Searches were evaluated with each heuristic.
- Statistics were measured for nodes generated, number of nodes traversed, and time of execution.

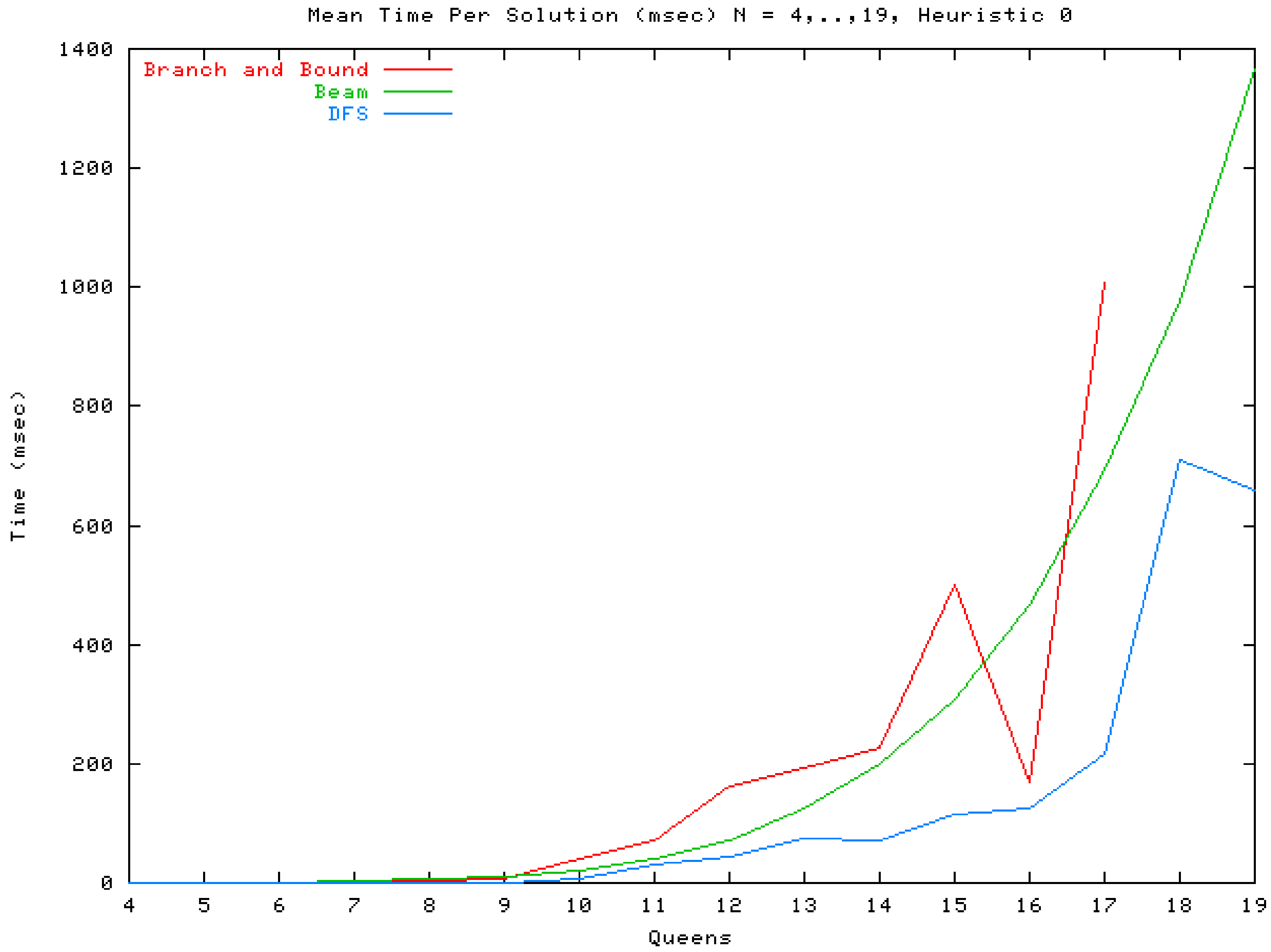
# Nodes Generated Per Solution, No Heuristic



# Nodes Traversed Per Solution, No Heuristic



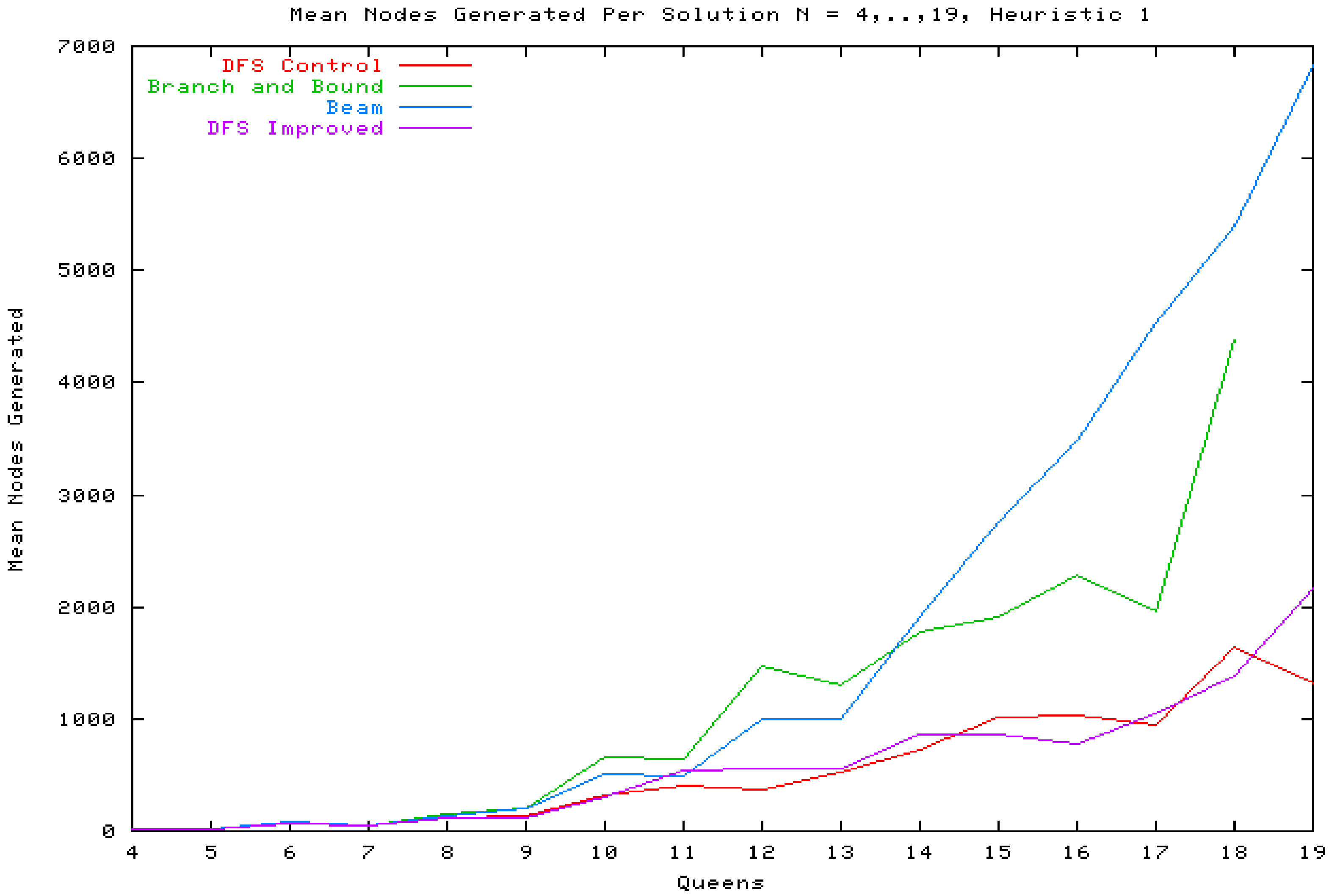
# Time to a Solution,  No Heuristic



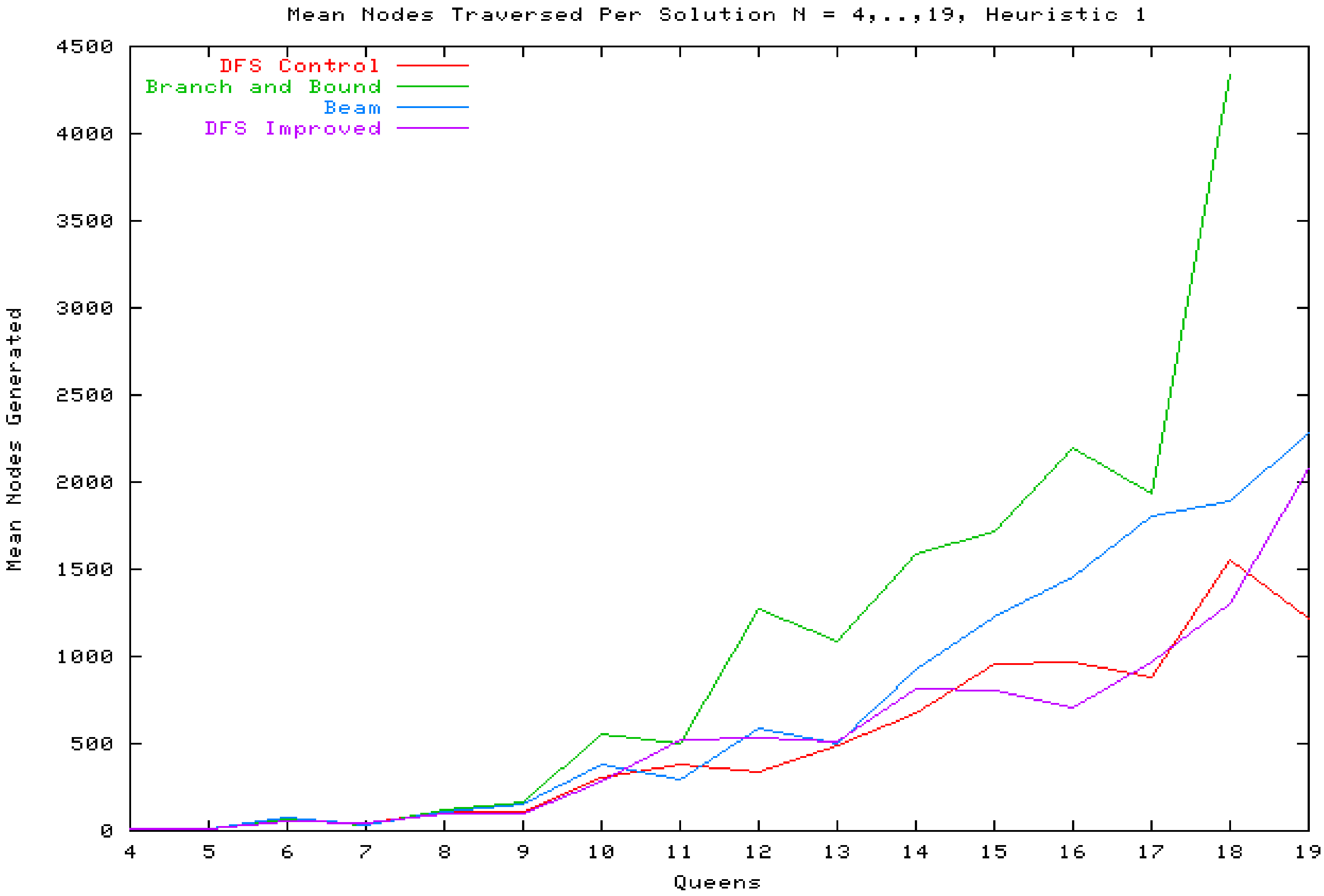
# Results: No Heuristic (H0)

- No Heuristic
- Works well with DFS.
- Poor performance and space efficiency with BnB and Beam Search.
- Had expected Beam Search to give mixed results and thought BnB would revert to breadth first search.

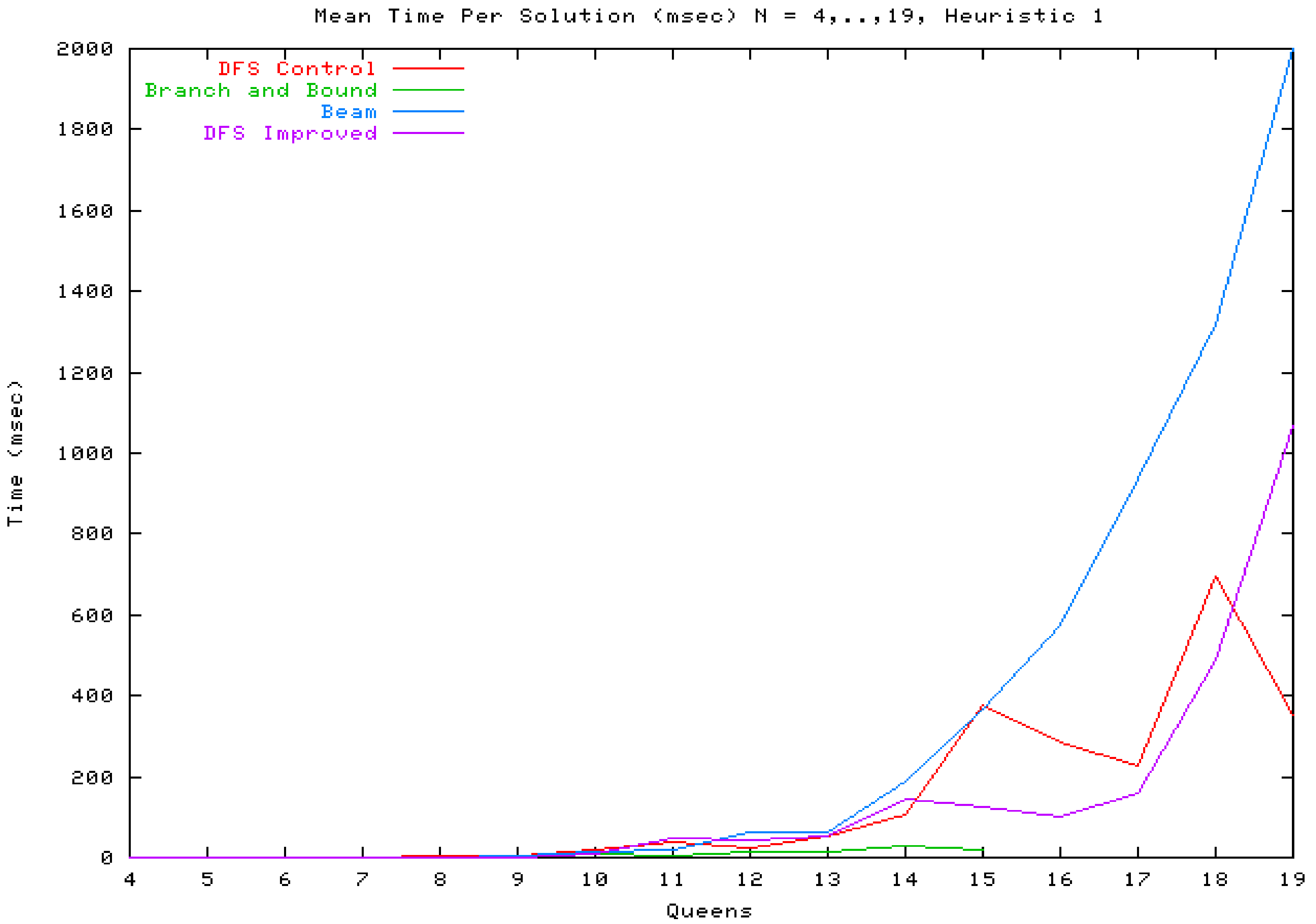
# Nodes Generated Per Solution, Distance From Previous Queen (H1)



# Nodes Traversed Per Solution, Distance From Previous Queen (H1)



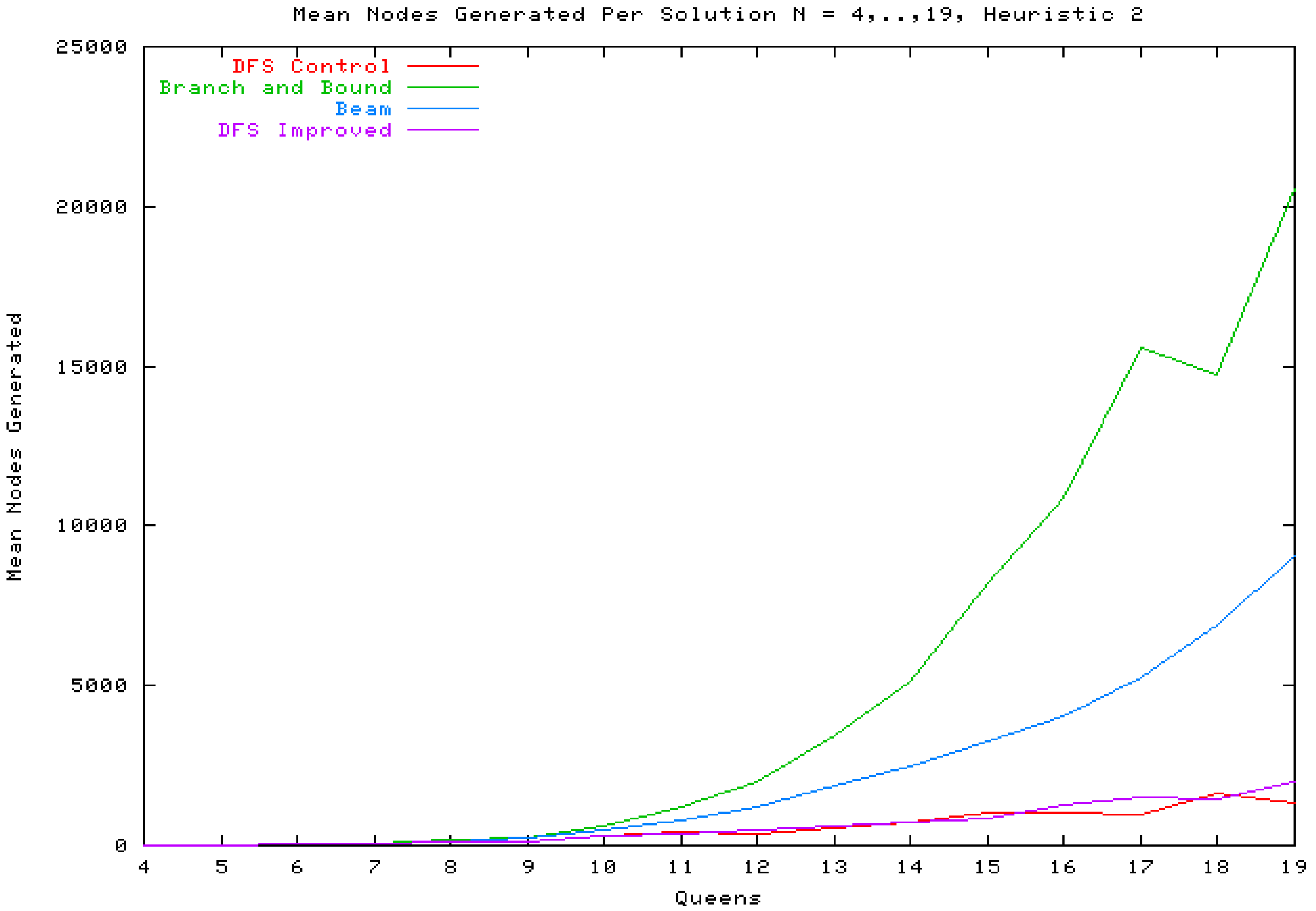
# Time to a Solution, Distance From Previous Queen (H1)



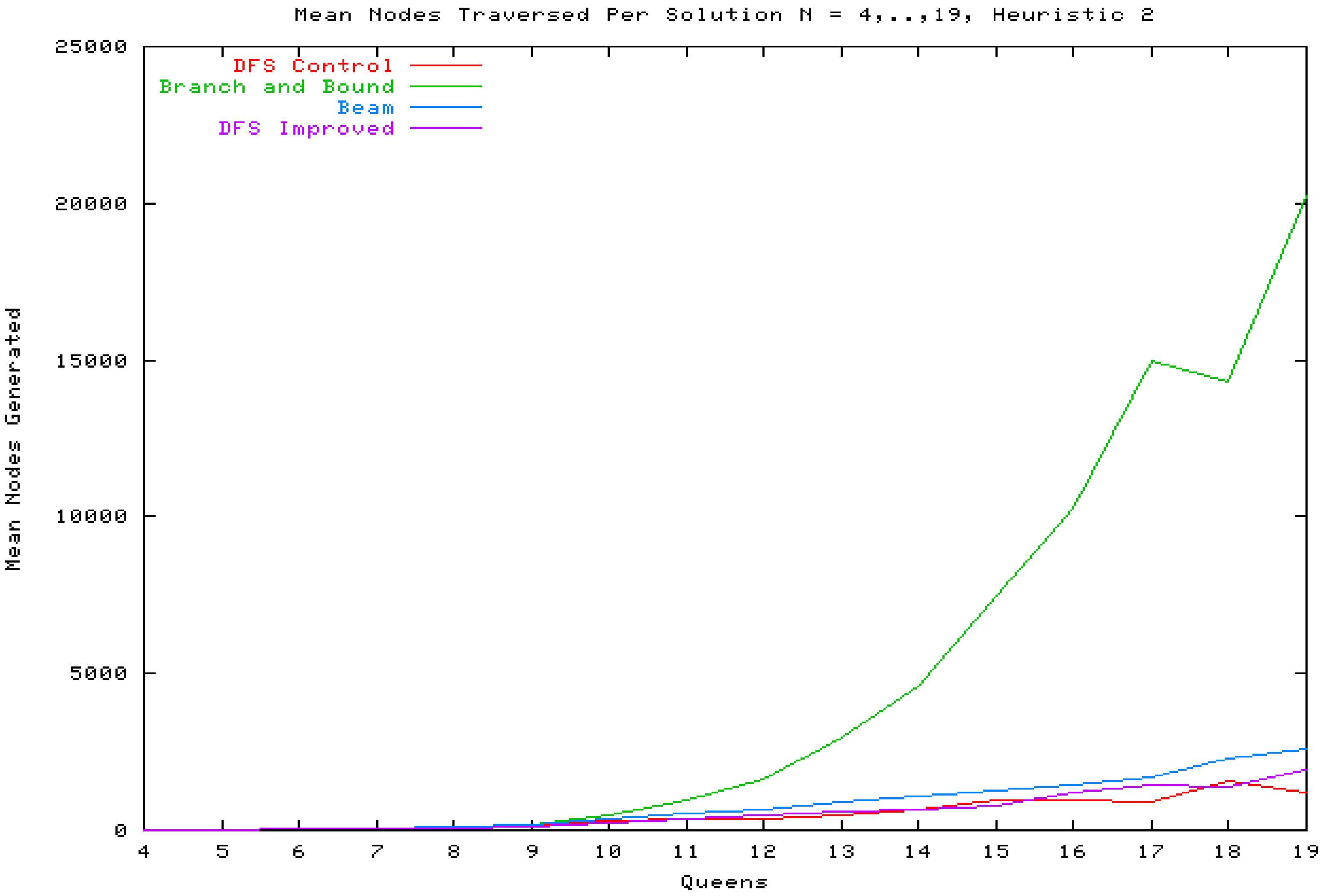
# Results: Simple Distance Measure (H1)

- BnB performs badly in most areas except time.
- Beam Search generates too much of the search space and takes too much time to compare well.
- DFS about same as improved DFS.
- Bad to worse

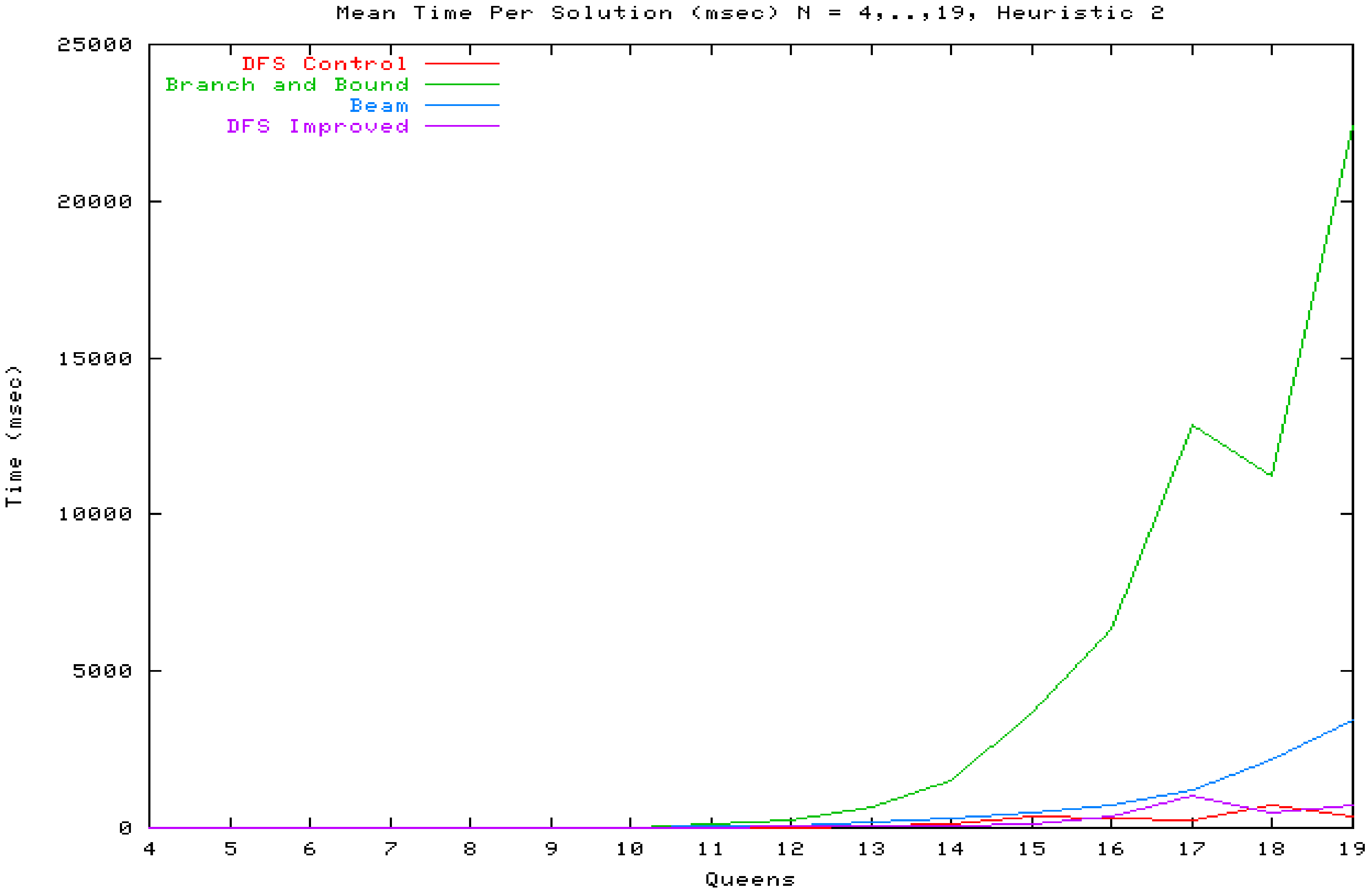
# Nodes Generated Per Solution, Mean Distance From Previous Queens (H2)



# Nodes Traversed Per Solution, Mean Distance From Previous Queens (H2)



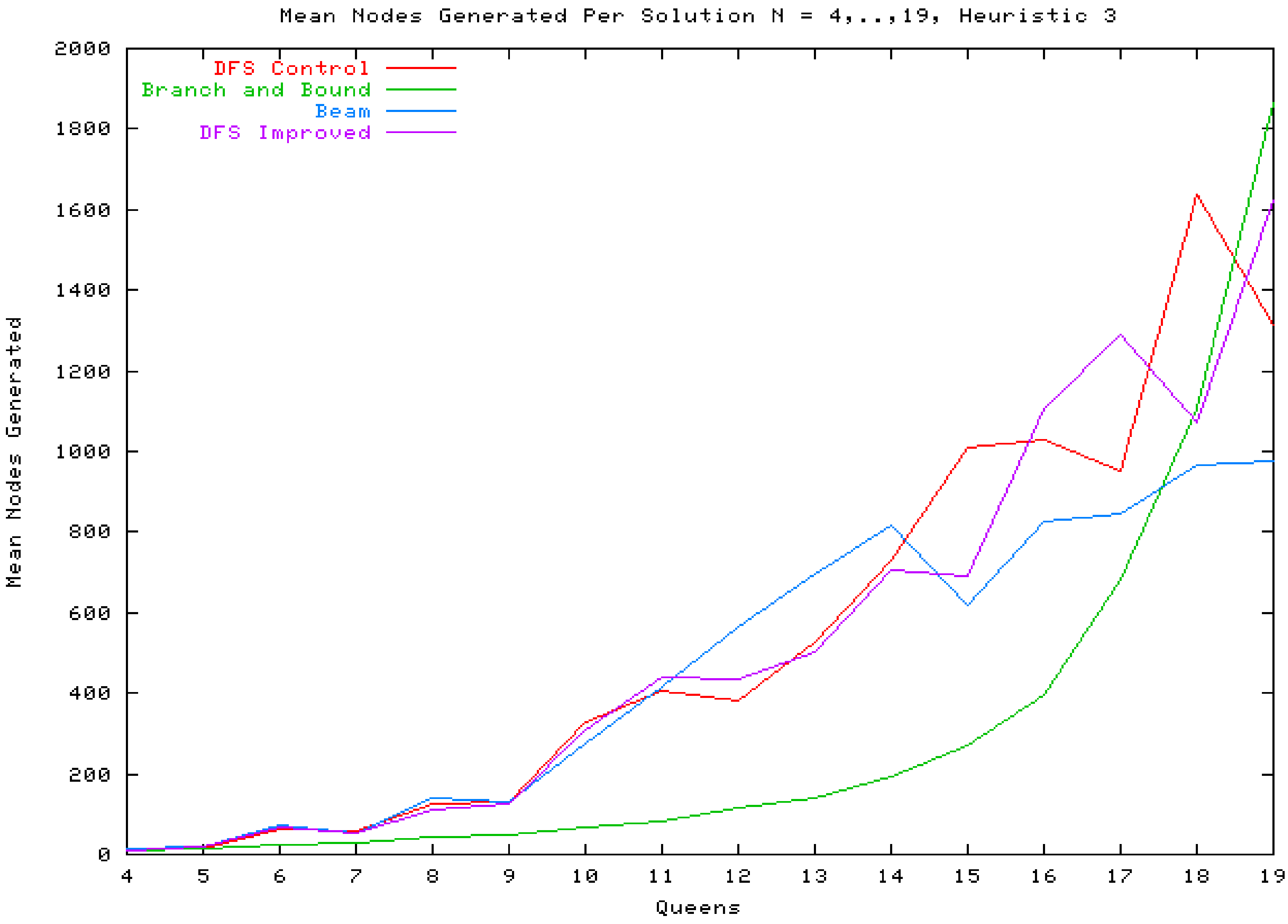
# Time to a Solution,  Mean Distance From Previous Queens (H2)



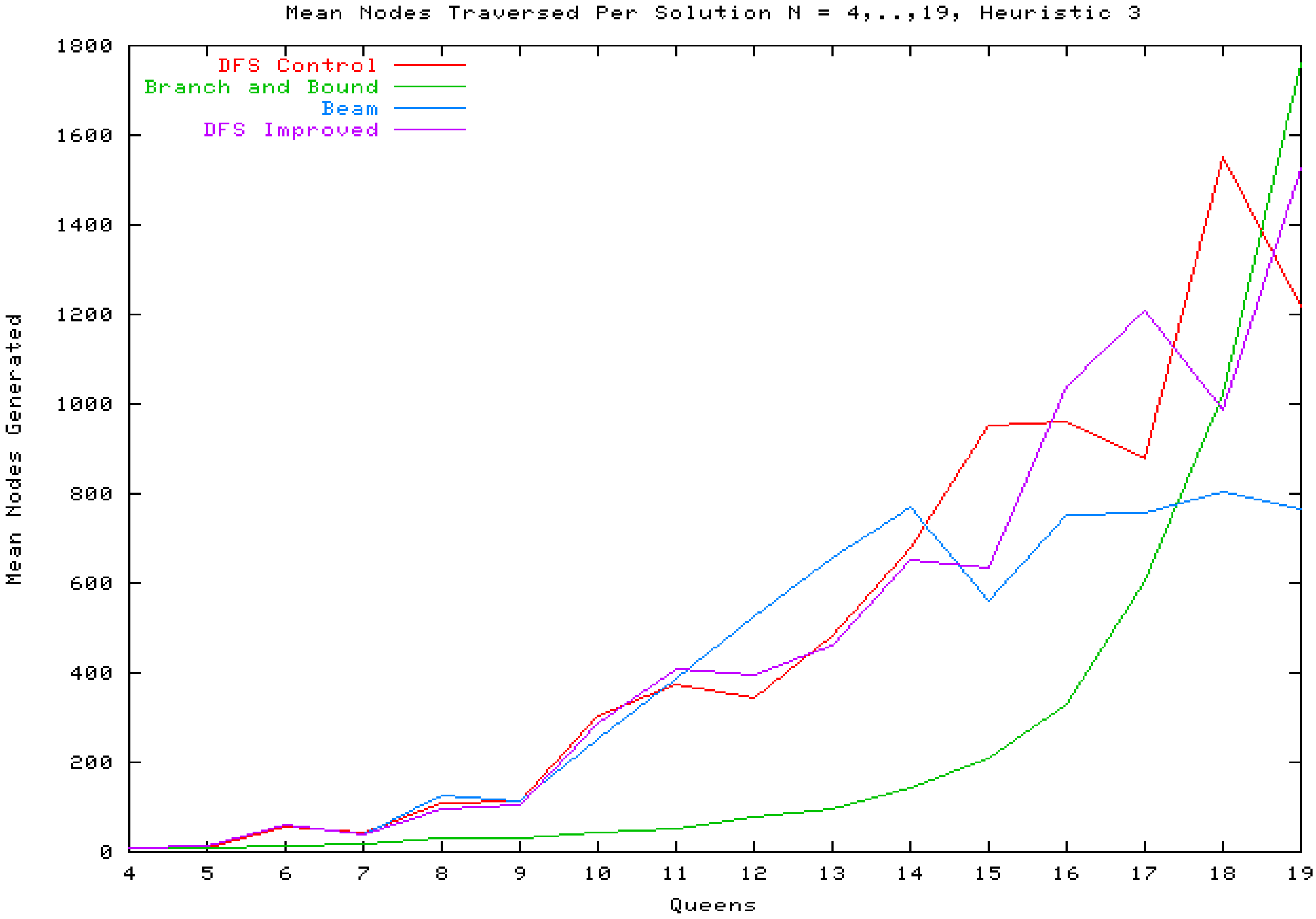
# Results: Aggregation of Simple Distance (H2)

- BnB performs badly in all areas.
- Beam Search is close to DFS in nodes traversed and time of execution, but generates too much search space.
- DFS still about the same as DFS improved.

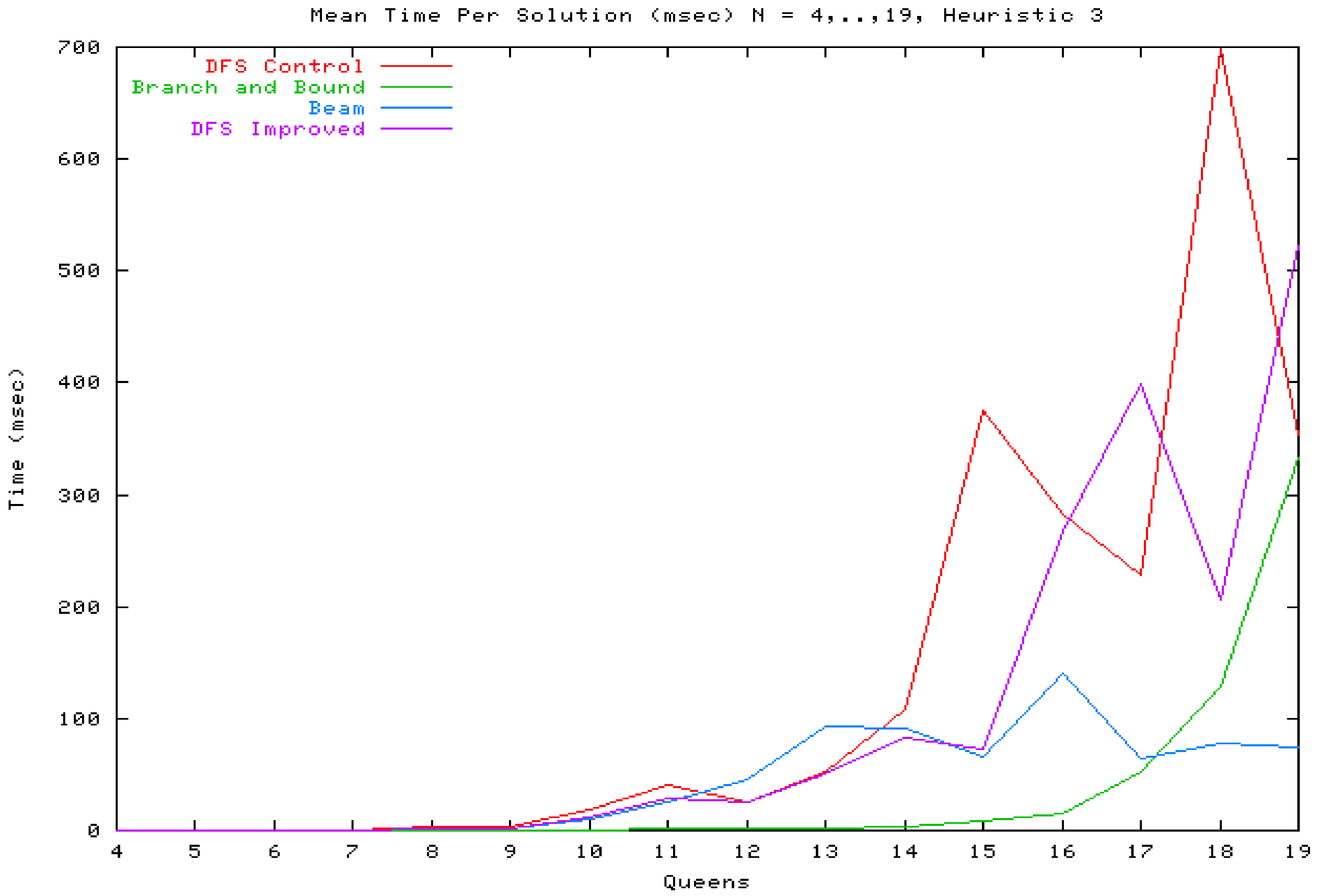
# Nodes Generated Per Solution, Free Squares



# Nodes Traversed Per Solution, Free Squares



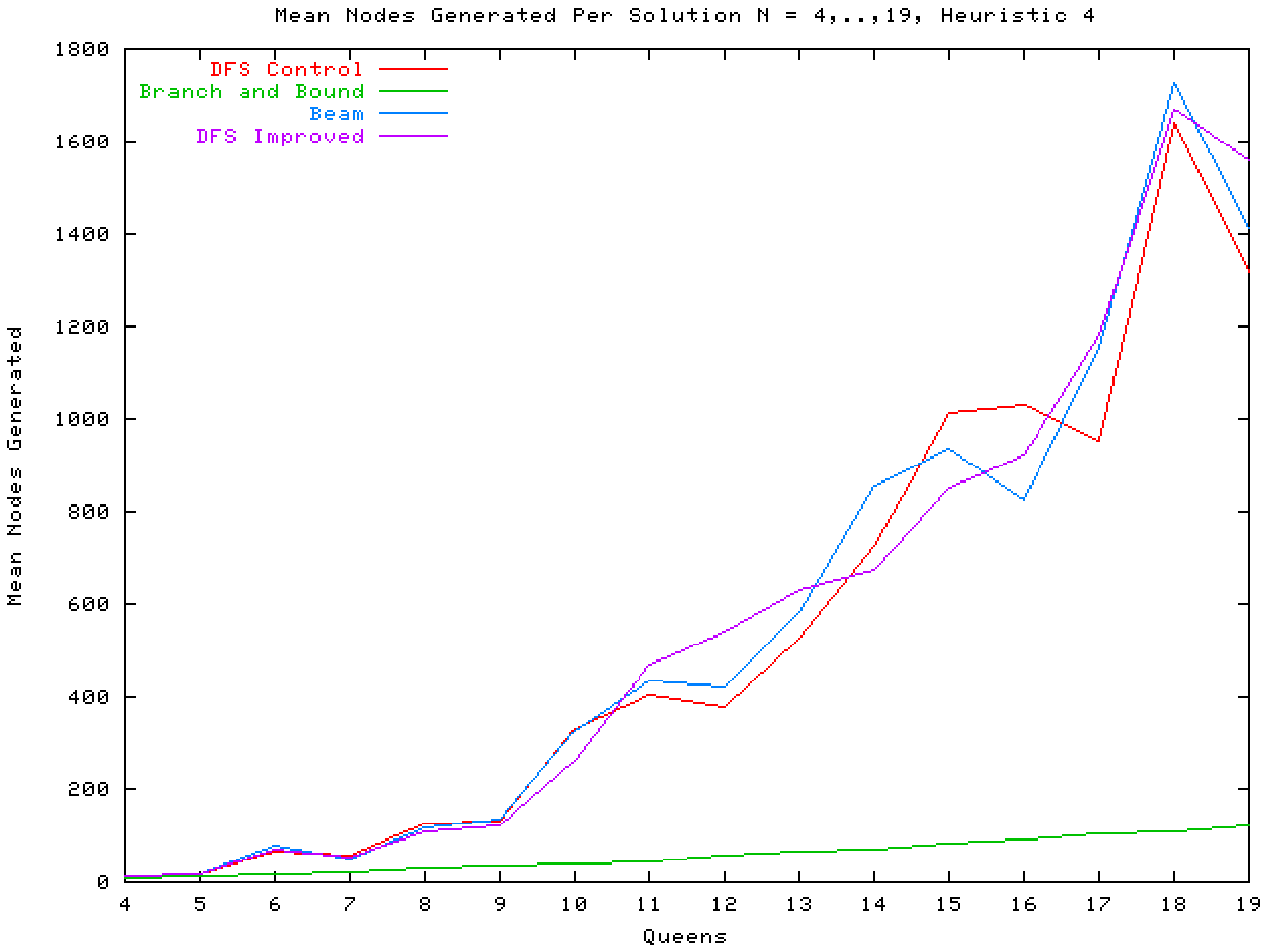
# Time to a Solution,  Free Squares



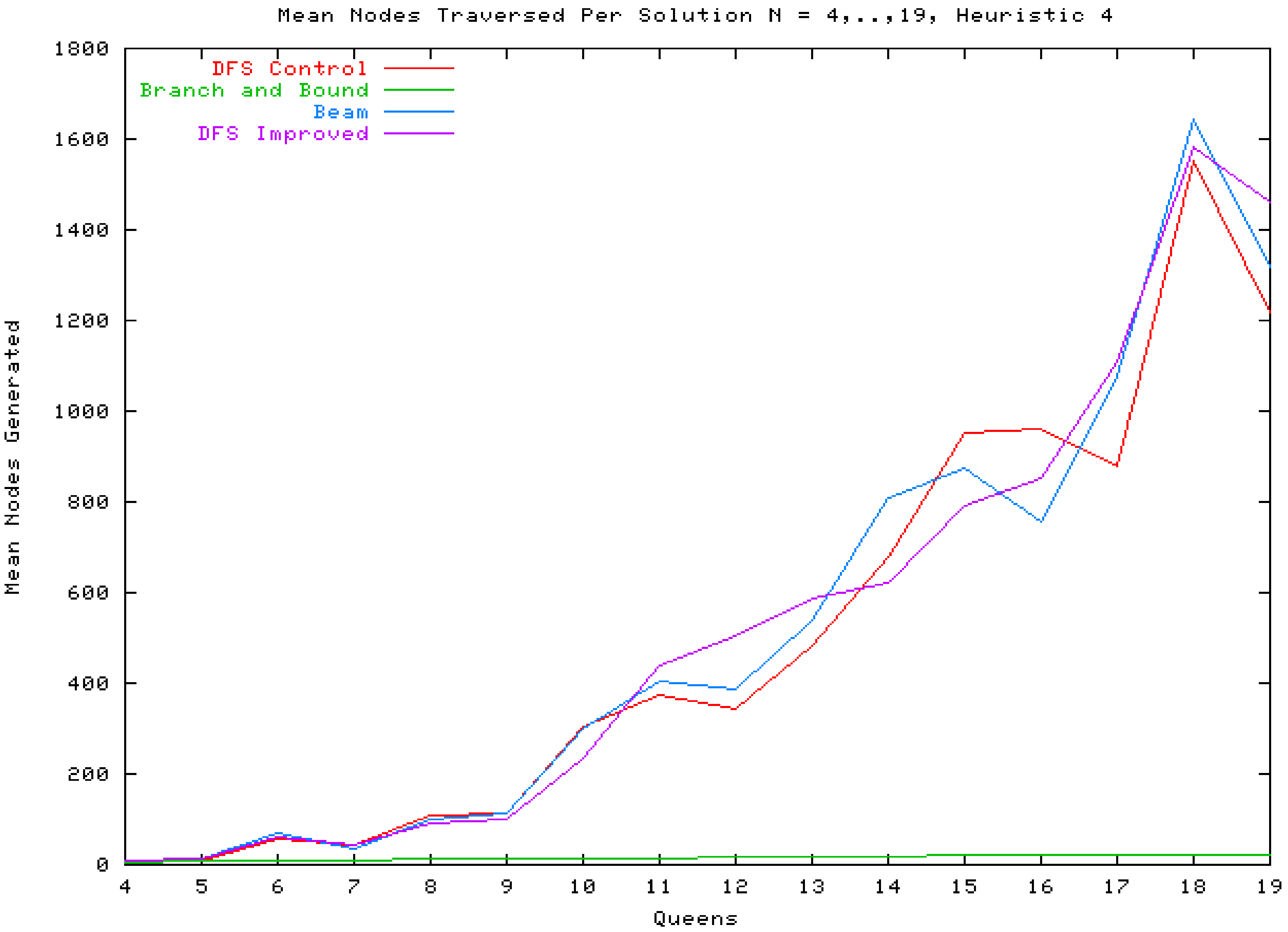
# Results: Number of Open Squares (H3)

- BnB, DFS, and DFS Improved all generated similar size search space and yielded similar node traversals and search times.
- Beam Search generated a third fewer nodes and traversed only half as many nodes on average to find a solution and executed in linear time.
- DFS still same as DFS improved.

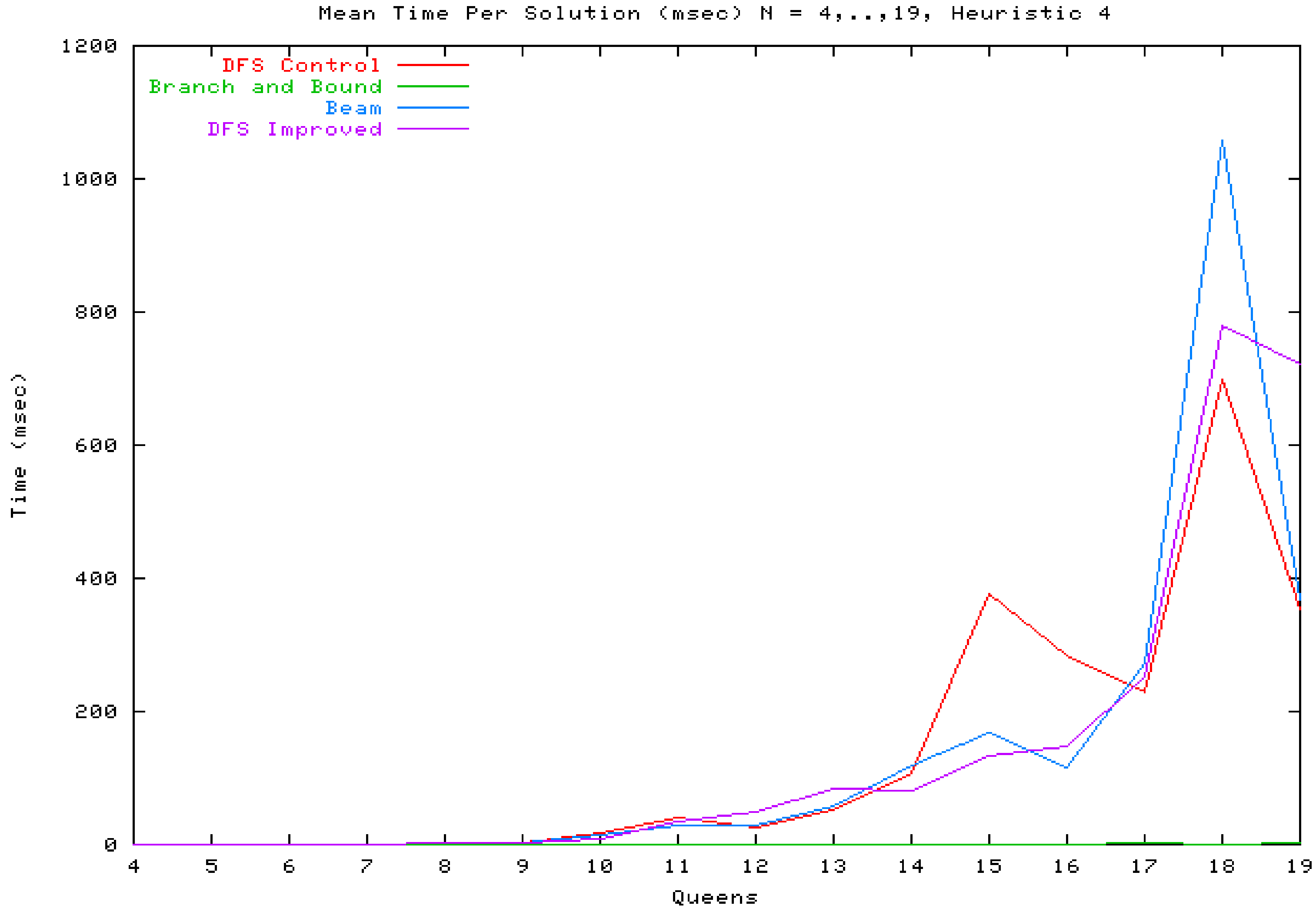
# Nodes Generated Per Solution, Mean Hamming



# Nodes Traversed Per Solution, Mean Hamming



# Time to a Solution,  Mean Hamming



# Results: Mean Hamming Disance (H4)

- Beam, plain DFS, and improved DFS all generated similar size search spaces and node traversals.
- Performance times for Beam Search and plain DFS were about the same. Improved DFS performed worse.
- BnB generated a compact solution space, traversed few nodes, was incredibly fast, when it generated a solution.

# Discussion

- Additional computation yielded a more informed search.
- Better heuristics may have interesting trade-offs, e.g. BnB used linear space, but did not always find a solution.

# Challenges

- Some paths permit no solution.
  - Even promising heuristic evaluations.
- Parameters must be tuned to match heuristics with searches.
- Results were difficult to evaluate.
  - Still finding significant information.

# Summary

- Selected heuristics work better with some searches than others.
- Beam Search matched well with the Open Squares heuristic.
- BnB performed well using the Mean Hamming distance heuristic.
- DFS Improved didn't perform much better than plain DFS. Waste of time.
- Heuristic Repair is looking better ...

# Selected References

- 1. Gauss's Arithmetization of the Problem 8 Queens, Ginsburg, January 1938, Scripta Mathmatica, Vol. 5, No. 1, Yeshiva College, New York.
- 2. Gauss and the Eight Queens Problem: A Study in Miniature of the Propagation of Historical Error, Campbell, Nov. 1977, Historia Mathematica, Vol. 4 No. 4.
- 3. The n-Queens Problem, Letavec and Ruggiero, May 2002, Informs Transactions on Education, Vol. 2, No. 3.
- 4. Different Perspectives of the N-Queens Problem, Erbas, Sarkeshik, and Tanik, 1992, ACM.
- 5. A Polynomial Time Algorithm for the N-Queens Problem, Sosic and Gu, 1990, SIGART, Vol. 1, No. 3.
- 6. Isomorphism and the N-Queens Problem, Cull and Pandey, Sept. 1994, SIGCSE Bulletin, Vol. 26, No. 3.
- 7. Minimizing Conflicts: A Heuristic Repair Method for Constraint-Satisfaction and Scheduling Problems, Minton, Philips, Johnston, P. Laird, 1993, Journal of Artificial Intelligence Research, Vol. 1.