TECHNISCHE UNIVERSITÄT DRESDEN

**Faculty of Computer Science** Institute for Computer Engineering

# Putting Queens in Carry Chains

**Thomas B. Preußer**

**Bernd Nägel** **Rainer G. Spallek**

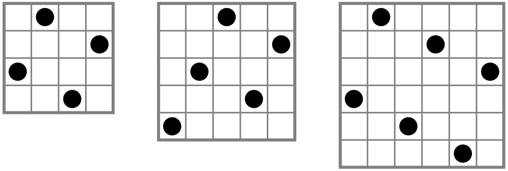
$\Pi\acute{\alpha}\phi o\varsigma$, HIPEAC WRC'09

## Itinerary

- Problem and Complexity Overview
- Solution Approach
- Hardware Mapping and Optimization
- Computation Status for $N = 26$

## The $N$-Queens Puzzle

- Place $N$ *non-attacking* Queens on a $N \times N$ chessboard:



- Generic solution templates?

$\longrightarrow$ How many (fundamental) solutions?

## Known Solution Counts

| $N$ | Solutions |
|---|---|
| 1 | 1 |
| 2 | 0 |
| 3 | 0 |
| 4 | 2 |
| 5 | 10 |
| 6 | 4 |
| 7 | 40 |
| 8 | 92 |
| 9 | 352 |
| 10 | 724 |
| 11 | 2680 |
| 12 | 14200 |
| 13 | 73712 |

| $N$ | Solutions |
|---|---|
| 14 | 365596 |
| 15 | 2279184 |
| 16 | 14772512 |
| 17 | 95815104 |
| 18 | 666090624 |
| 19 | 4968057848 |
| 20 | 39029188884 |
| 21 | 314666222712 |
| 22 | 2691008701644 |
| 23 | 24233937684440 |
| 24 | 227514171973736 |
| 25 | 2207893435808352 |
| 26 | ? |

Exhaustive backtracking solution exploration scales **overexponentially** with $N$.

Hard beyond $N = 20$.

<u>$N = 25$:</u>

- Java grid computation by INRIA, France.
- Runtime:

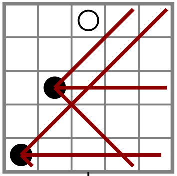
|  |  |
|---|---|
| Real | >6 Months |
| Sequential | >53 Years |

## Tackling $N = 26$

- Embarrassingly parallel workload:
  1. Preplace $L \ll N$ columns.
  2. Explore subboards **independently**.
  3. Collect and add up subtotals.
- Ideally suited for distributed computation:
  - Internet (BOINC) $\longrightarrow$ NQueens@Home
  - FPGA! $\longrightarrow$ Queens@TUD
    - Challenge the power of a world-wide distributed computation effort by an intelligent FPGA implementation.
    - Identified and reported an overflow bug on November 7, 2008.
    - Thereby, resolved an open dispute on the solution for $N = 24$ *without* any own computation.

## Algorithmic Overview

Exhaustive backtracking solution exploration.



valid placement yet to explore?

Yes

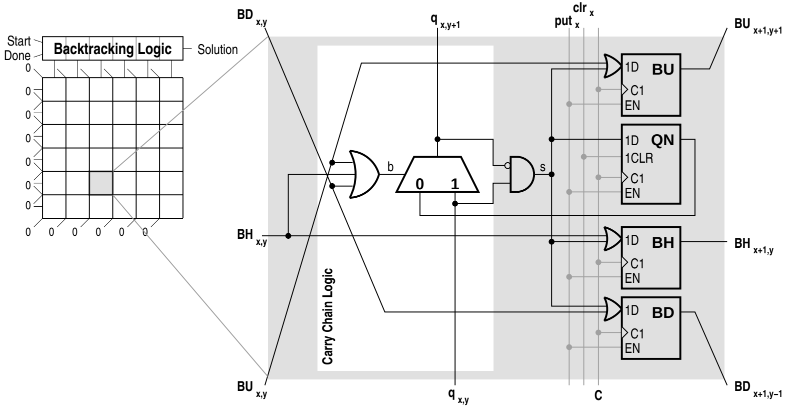
1. Mark *explored.*
2. Update blocking.
3. Advance to next column / Count solution.

No

1. Clear Markings.
2. Retreat to previous column / Done.

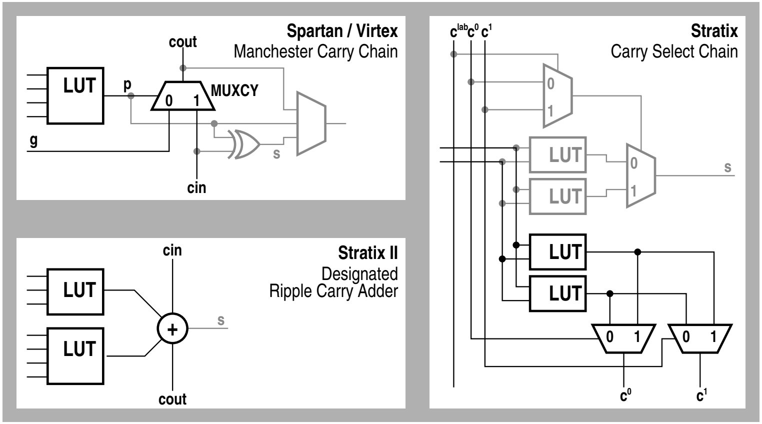
Calculated *Blocking Vectors* avoid frequent constraint validation.

## FPGA Mapping



TECHNISCHE UNIVERSITÄT DRESDEN

## Carry Chain Structures



## Generic Carry-Chain Mapping through Addition

1. **Derive Carry / Token Propagation**

| Case | $c_{i+1}$ | Description |
|---|---|---|
| $k_i$: Kill | $0$ | never a carry: holding no queen and not blocked |
| $p_i$: Propagate | $c_i$ | pass a carry: holding no queen and blocked |
| $g_i$: Generate | $1$ | always a carry: holding current queen placement |

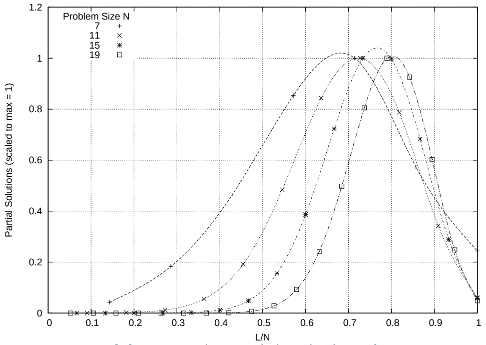
2. **Determine Addends**

$$\left.\begin{array}{rcl} a_i & = & g_i + p_i \\ b_i & = & g_i \end{array}\right\} \text{ always valid, possibly suboptimal}$$

3. **Fix up Sum Bits of $s <= a + b$**
   In equations dependent on the incoming carry / token use:

$$c_i = s_i \oplus p_i$$

## Partial Solutions Development



Many *stale* partial solutions!

## Partial Solutions – Consequences

- Partition problem with $L < \frac{N}{2}$.
  - Otherwise the subproblems might outnumber the actual solutions, thus, producing a large communication overhead.
  - We chose $L = 6$ for $N = 26$.
- *Early* identification of stale search trees might be valuable.
  - Our attempts projecting the current blocking vectors onto a single or onto pairs of somewhat distant future columns only produced a disappointing cycle reduction.

## Pushing Performance

Optimization for a small size and a high clock frequency:

- Retain a single active column (without fast token handback).
- Keep placed columns within a plain array of shifted registers.
- Use global blocking vectors for all rows and diagonals setting and unsetting placements and retreats, respectively.
  Note that this would be too expensive in software!

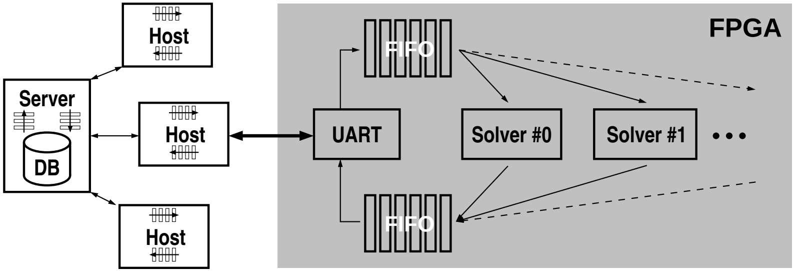
Outcome:

- Cycle count to finish a subboard more than doubled.
- Size reduced to nearly a quarter.
  → This also improved device utilization.
- Clock frequency grew by more than 50%.

→ Performance tripled.

TECHNISCHE UNIVERSITÄT DRESDEN

## Project Infrastructure



TECHNISCHE UNIVERSITÄT DRESDEN

## Contributing Devices

| Nmbr. | Device | Solvers | Clock | Perf. |
|---|---|---|---|---|
| 4× | Spartan-3 XC3S1000 | 15 | 92 MHz | 13.8 SE |
| 2× | Spartan-3E XC3S500E | 9 | 100 MHz | 9.0 SE |
| 1× | Stratix EP1S80 | 74 | 120 MHz | 88.8 SE |
| 1× | Stratix II EP2SGX90 | 78 | 160 MHz | 124.8 SE |
| 4× | Virtex-5 XC5VLX50T | 33 | 163 MHz | 53.8 SE |
| AMD Quad-Core Phenom(tm) 9850, Stepping 3 | | 4 | 2.5 GHz | 8.0 SE |
| Intel Core2 Duo CPU, Stepping 11 | | 2 | 2.7 GHz | 3.6 SE |

SE – Slice Equivalent measured as performance of a single 100 MHz slice.

- Even small FPGA devices easily outperform a workstation CPU.

TECHNISCHE UNIVERSITÄT DRESDEN

## Computation Status

Constantly published on project website: `http://queens.inf.tu-dresden.de`

- Computational Progress (currently >21% of 25204802 subboards).
- Subtotals.

Thank you!