

# WaferConnect™

The DreamWafer's software

Etienne Lepercq

May 29, 2009

# The DreamWafer's software

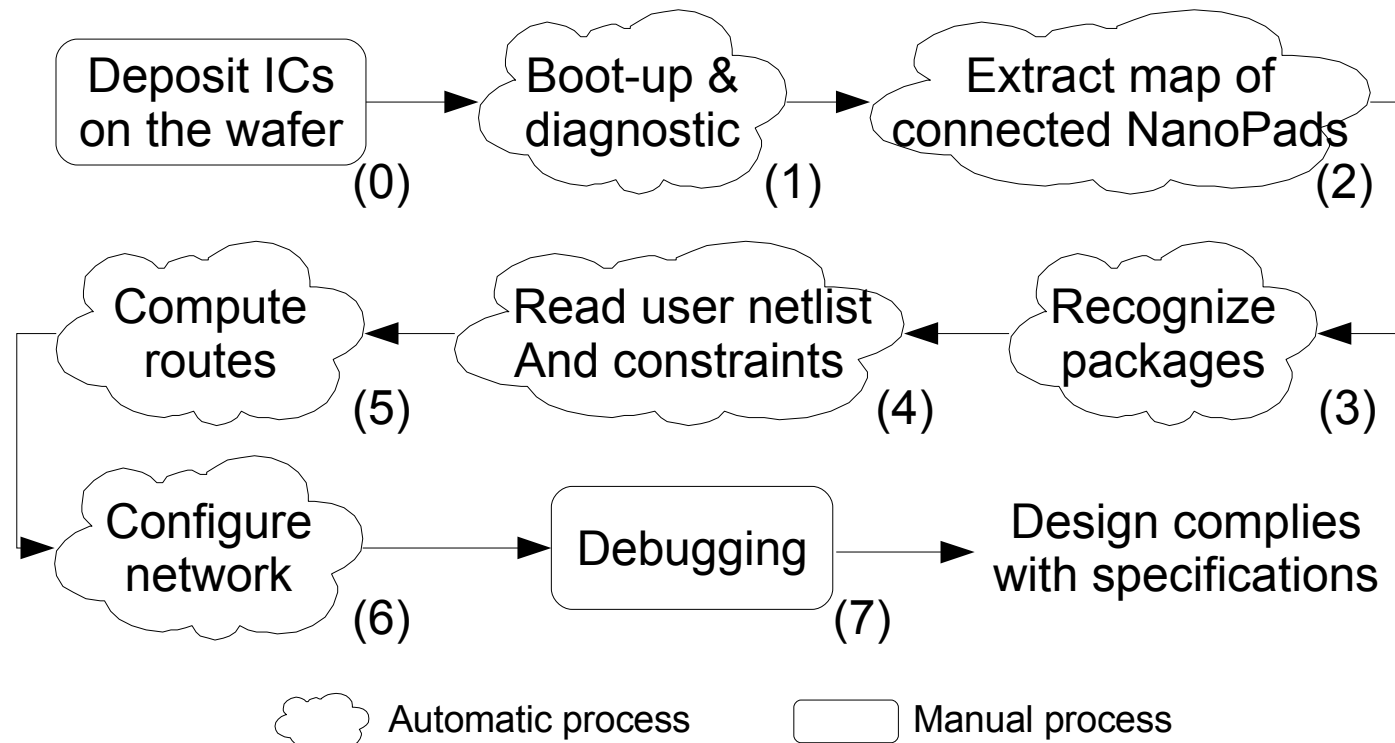
- Goals and purpose
  - Interface between user and the WaferBoard
  - Accelerate development
    - Minimize manual processes
    - Algorithms must be fast enough to improve user experience
  -

# The DreamWafer's software

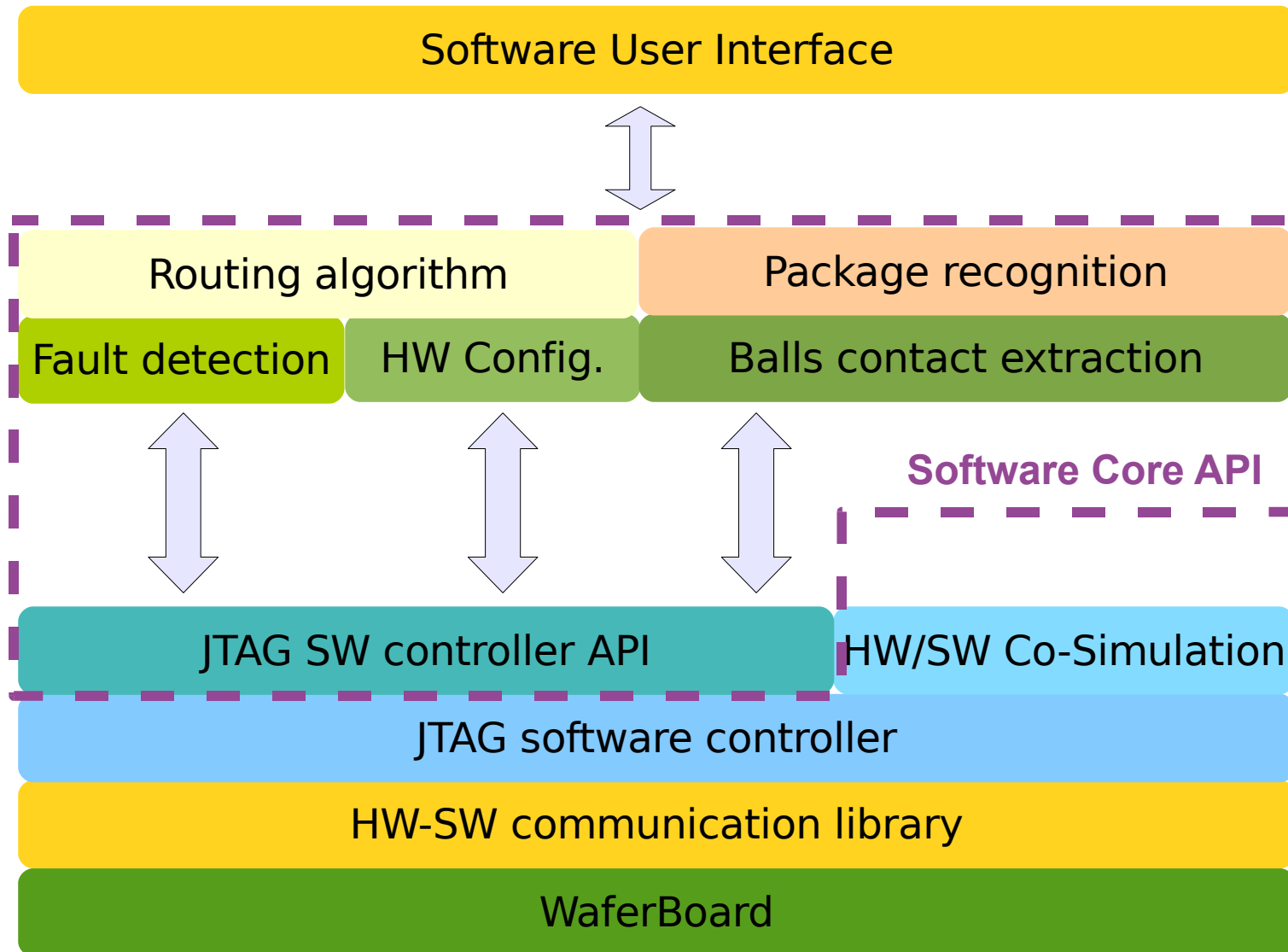
- Most important building blocks
  - Communication module (USB/PCI-Express/...)
  - Software JTAG controller
  - Fault detection & Boot-up procedures
  - Balls contact extraction
  - Package recognition
  - Routing algorithm
  - HW configuration
  - Software Core API
  - User interface

# The DreamWafer's software

- Software workflow - proposed

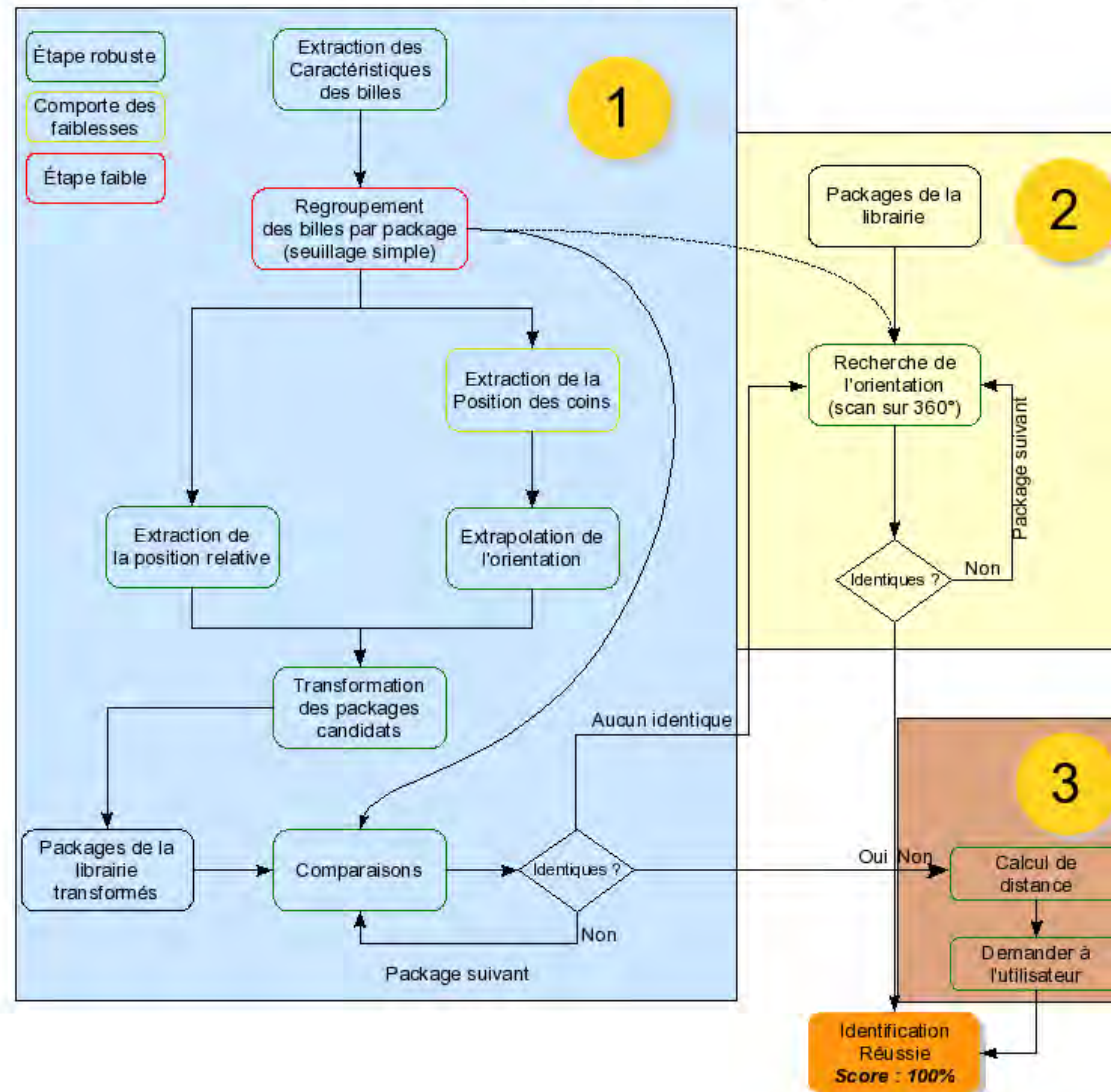


# The DreamWafer's software



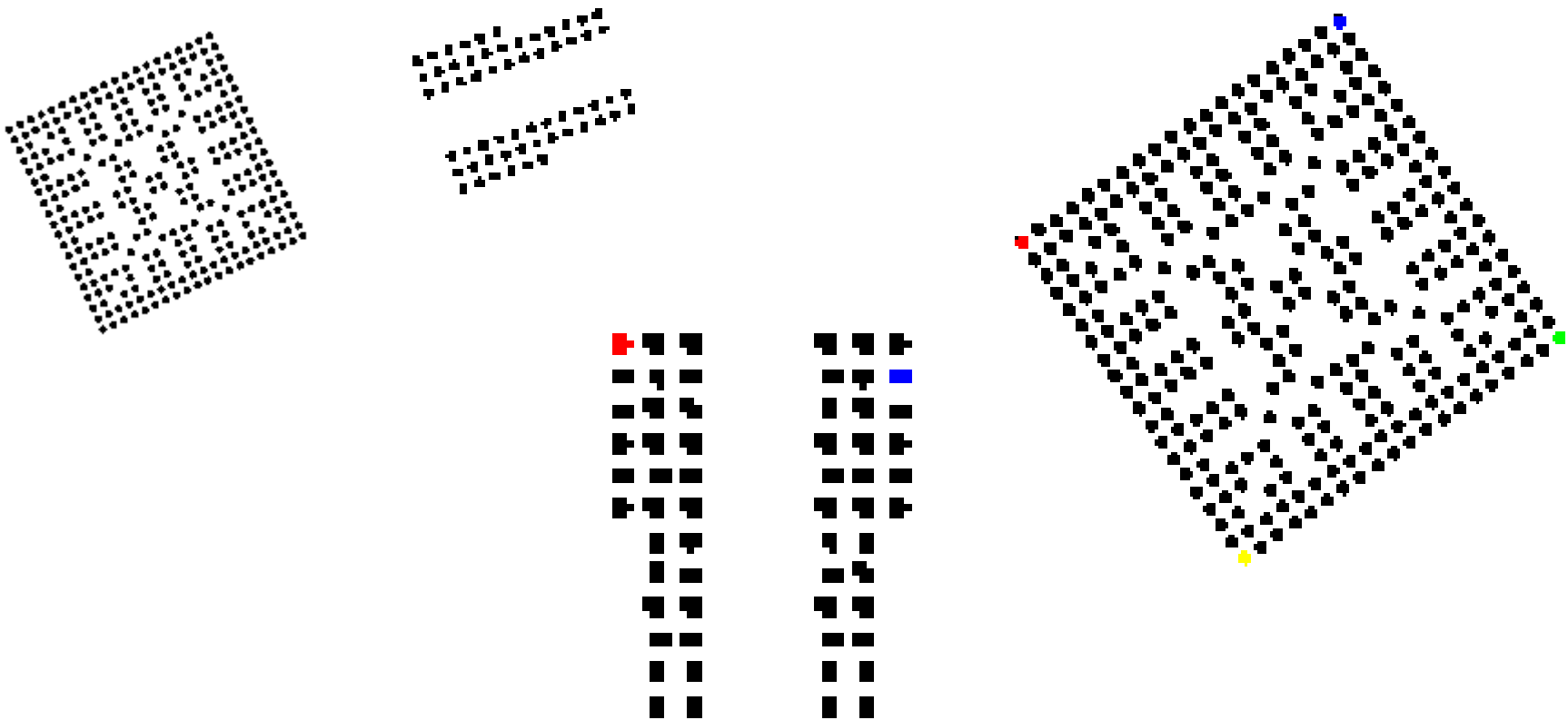
# Package recognition algorithm

Reconnaissance de package - algorithme



# Package recognition algorithm

- Benchmark examples and corner extraction



# Routing algorithm

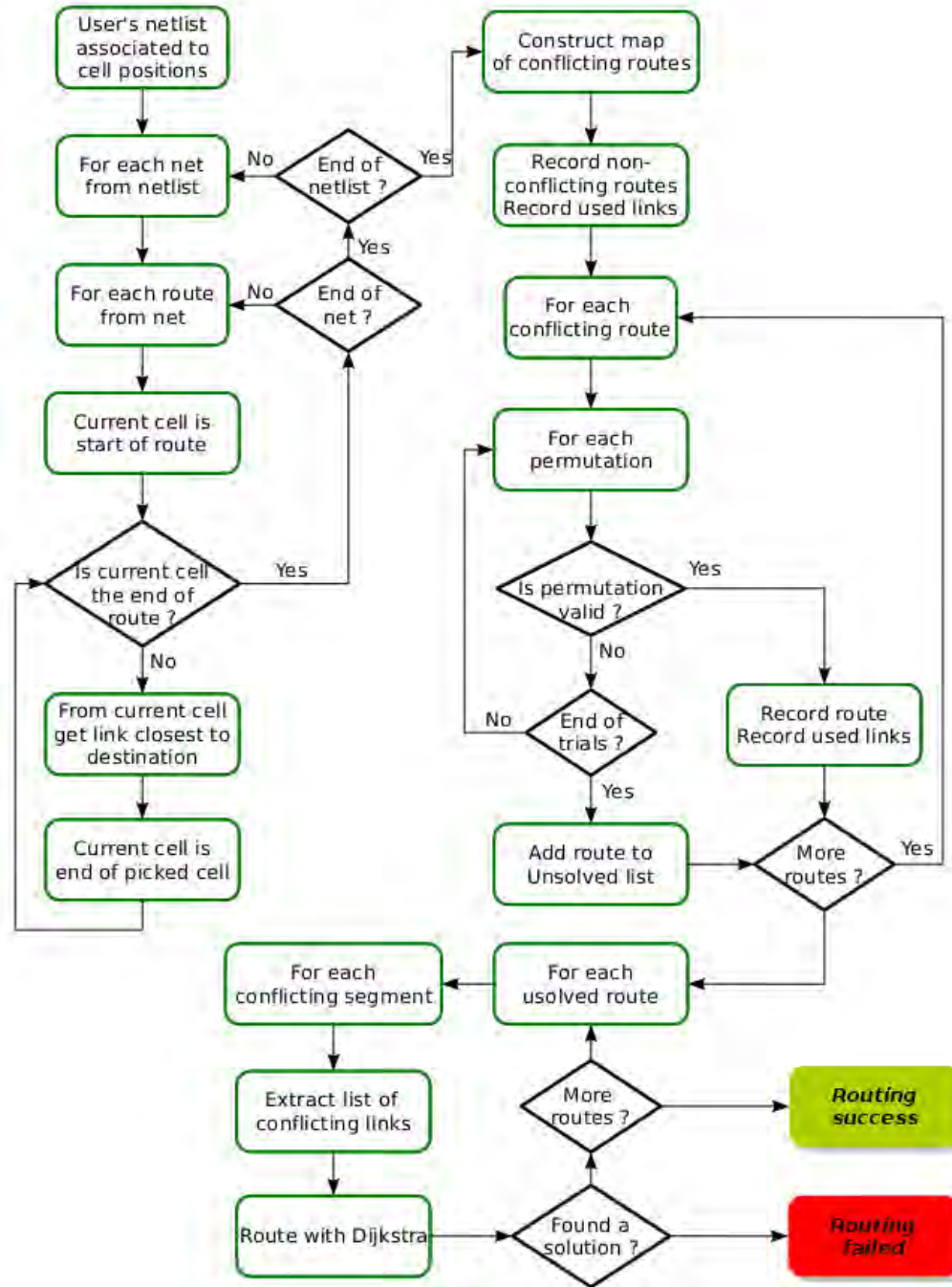
- Finding non conflicting k-minimum routes is a NP-hard problem
  - Size of the problem (~2.5 millions vertices) make it impossible to find an optimal solution
  - Problem settle at 1/3 close to an FPGA routing problem and 2/3 of PCB routing problem
  - Using heuristics and leveraging WaferNet's architecture is mandatory to route big netlists in short times (~minute)



# Routing algorithm

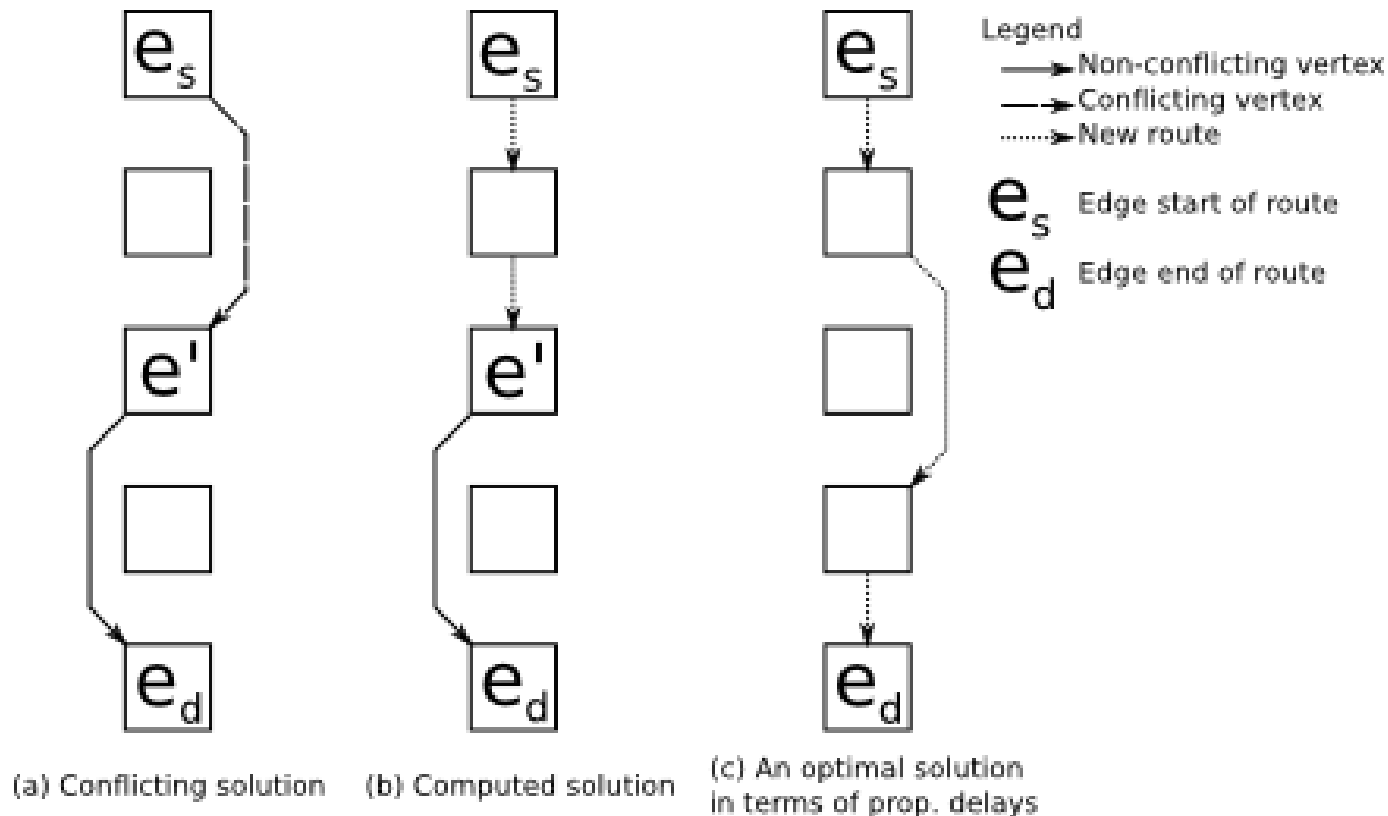
- Data architecture:
  - A set of nets (think VDD, GND, VDD3.3, CLK, ...)
  - A set of routes (think a set of balls connected to same net)
- Two problems:
  - For each route, find a path non-conflicting with others
  - For nets with high degree, find a minimum spanning-tree (another NP-hard problem !)

# Routing algorithm



# Routing algorithm

- Dijkstra's rerouting – a more efficient solution



# Routing algorithm

- A very quick overview on results
  - Able to route very dense netlists on real-sized wafer
  - A few seconds to compute a 5-10% dense netlist
  - Complexity starts to hurt when reaching 20%+ density on real-sized wafer and long routes
- There is large room for optimization (or fun), but we reached an acceptable performance :-)
- Working on NP-hard problems is interesting