Dependable multiprocessors

- **Development of technology $\Rightarrow$ reliability of components $\Uparrow$**
- **Sophisticated equipment (more components) $\Rightarrow$ reliability of equipment $\Downarrow$**
- **High degree of dependability**
  - **Large scale multiprocessors**
  - **Mission critical applications**
    - **High data volumes**
    - **Long mission time**
    - **Safety-relevance**
  - **Scalability**
  - **Moderate redundancy $\Rightarrow$ fault tolerance:**

Categories of parallel data processing systems

- **Multi-computers**
  - Distributed Systems (DS)
  - Workstation Clusters (WSC)
  - Massively Parallel (message based) Multi-computers (MPP)
- **Symmetric multi-processors (SMP)**
  - Global shared memory
  - Conventional programming models
  - Not scalable
  - $\leq n \times 10$ processors
- **Symmetric multiprocessor clusters**
  - Symmetric multiprocessors or super-workstations
  - Clustering via high speed interconnection networks.
  - High performance per node
  - Scalability

Fault tolerance

- **Ability of a system to**
  - ***tolerate*** **a bounded number of faults and resulting errors**
  - ***continue*** **its operation until scheduled maintenance**
- **Requires redundancy in**
  - **hardware**
  - ***software***
  - ***time***
- **Techniques**
  - **Fault masking (expensive hardware redundancy**
  - ***Fault handling***

Phases of fault tolerance

- **Error detection and fault localization (diagnosis)**
- **Error containment**
- **Fault treatment**
  - **E.g. fault isolation and error recovery**
  - **Checkpoints and roll-back (backward error recovery)**
  - **Time redundancy**

Fault handling mechanisms

- **Multi-computer**
  - **Error detection: *monoprocessor* techniques**
    - **E.g. self-test or self-checking logic**
    - **System level diagnosis: *distributed***
  - **Error containment region: processing node**
    - **Error propagation: erroneous messages**
  - **Fault treatment:**
    - **Isolation of the faulty processing node**
    - **Checkpointing with roll-back (messages replayed)**

- **Multi-processor**
  - ***Concurrent* error detection**
    - **Lock-step result comparison (master-checker)**
    - **Behavioural monitoring (watch-dog processor)**
  - **Error containment region?**
    - **Error propagation: unconstrained (state sharing by processors)**
      - **Shared variables**
      - **Cache snooping**
  - **Fault treatment:**
    - **Identification of process interaction (What parts of memory saved at a checkpoint?)**
    - **Isolation of faulty components**

Typical fault tolerance mechanisms

<table>
<tr><th>Phase of fault handling</th><th>Distributed system</th><th>WS cluster</th><th>SMP</th><th>Clustered SMP</th><th>MPP</th></tr>
<tr><td rowspan="3">Error detection (proc.)</td><td colspan="5">Monoprocessor error detection
- Hardware assisted (MMU, time-out watch-dog)
- Self-test based</td></tr>
<tr><td colspan="2">Application specific
- Transactions
- Assertions</td><td colspan="2">- Master-checker setup
- Watch-dog processors
- Off-line tests</td><td>Diagnostic con-trol network</td></tr>
<tr><td colspan="2">- I’ am alive messages</td><td>- ECC
- Scrubbing</td><td colspan="2">Shared checker HW
- watch-dog</td></tr>
<tr><td>Error detection (inter-connection)</td><td colspan="2">- Software time-out
- Protocol check
- ECC</td><td>- ECC</td><td></td><td></td></tr>
<tr><td>Damage confinement</td><td colspan="2">- Fail fast
- Fail soft
- Off-line diagnosis</td><td>- Fault masking</td><td>- System level diagnosis
- Fault isolation</td><td>- Checkpoints</td></tr>
<tr><td>Recovery</td><td colspan="3">- Shadow processes
- Degradation
- Reconfiguration
- Exception handling
- Forward recovery
- Redo</td><td colspan="2">- Redundant communication topology
- Redundant nodes
- Backward recovery</td></tr>
</table>

Error detection mechanisms

- **Compromise between overhead and dependability**
  - **Error-free checkpoint data, avoidance of error dissemination ⇒ High fault coverage , low error latency ⇒hardware based error detection**
  - **Replication of local hardware ED mechanisms ⇒ overhead**
  - **Limitations of software based redundancy:**
    - **Common mode faults affecting both the application and checking**
    - **Large run-time overhead**
- **Constructions used in reliable multiprocessors.**
  - **Structure based fault tolerance ⇒ Reliable hardware by general-purpose EDC mechanisms**
  - **Monoprocessor techniques in each node: computing core, control flow**
    - **MMU**
    - **Master-checker**
    - **WDP**
  - **Fault-tolerant interconnection network (redundant topology)**
  - **Global checking mechanisms for the entire multiprocessor**
    - **Diagnostic processor**
    - **Diagnostic resource sharing**
- **Algorithm-based fault tolerance: redundant implementation.**
  - **Redundant encoding of data structures**
  - **Application-dependant semantic constraints**

Validation of error detection mechanisms

- **Fault statistics acquisition and a posteriori evaluation**
- **Analytical methods (reliability analysis)**
  - Limitations in model complexity and size
  - Transient errors (reproducibility)
- **Simulation**
  - Detailed system and fault models
  - Huge modelling efforts.
- **Fault injection based experiments**
  - Interaction of nodes: simple software fault injectors at the protocol level.
  - Monoprocessors: hardware fault injectors emulating of the effects of physical faults
    - Heavy-ion radiation
    - Pin-level signal inversion
  - Software fault injectors: error level (pseudorandom CPU register or memory cell bit errors)
    - Correlation of physical and modelled fault effects
    - Dependence on the benchmark

Diagnostics approaches

- Requirement: high probability + speed
- Approaches:
  - Topology driven diagnosis
    - Starting point: topological location of the error
    - Rough fault model
    - Granularity: whole cluster or computing node (e.g. system-level diagnosis)
    - High level of diagnostic uncertainty
    - Pessimistic diagnosis
    - Improper modelling of communication components (routers, shared memories etc.)
- Unit-level diagnosis
  - Aim: immediate preparation of
    - reconfiguration
    - degradation
  - Level: components in reconfiguration (processor, link, router)
  - Fine granular models, computational complexity problems
- Functional diagnosis
- Aim: determination of fault duration (or)
  - Permanent: repair
  - Transient: recovery
- Hardware-oriented (self-) test programs as part of error handling