System Modeling and Software Implementation of MPEG-4 Encoder

Chen He

Shi Zhong

May 4, 2000

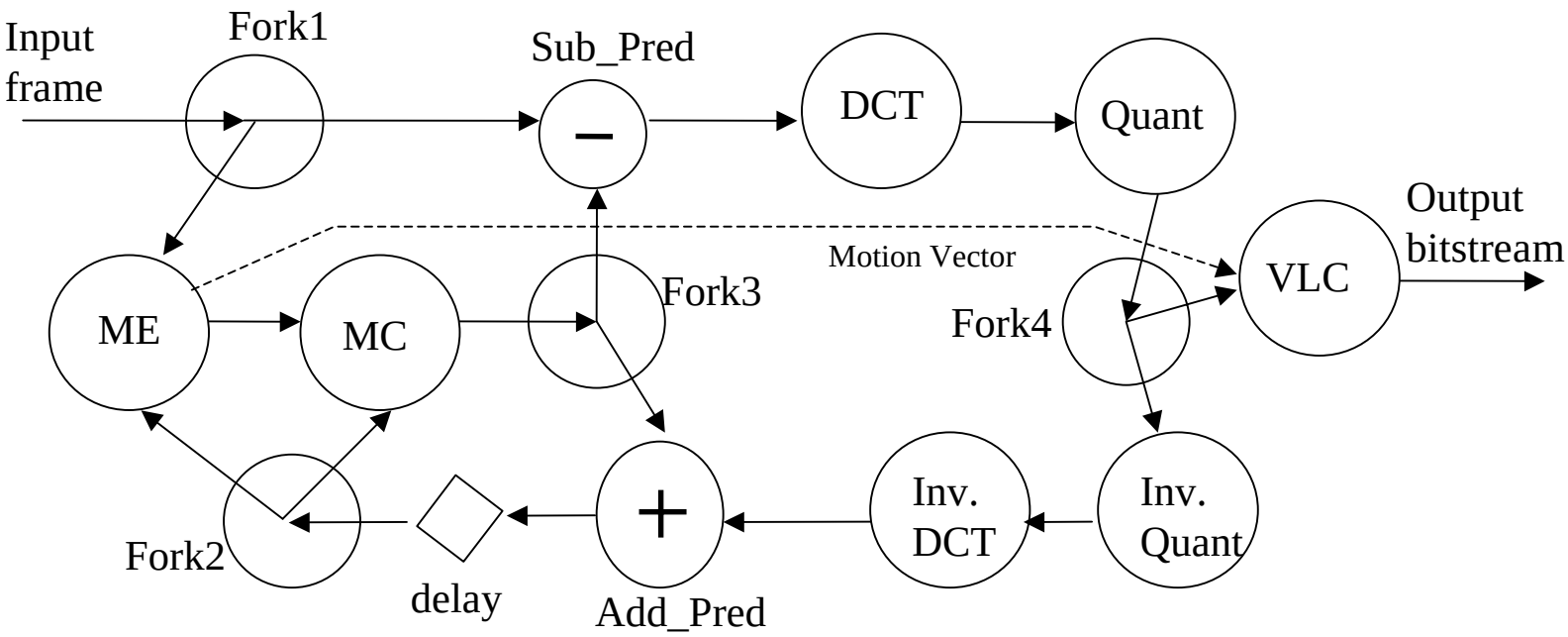
Problem Statement

- Real-time implementation of MPEG-4 encoder
  - Computation-intensive
  - Inherent parallelism
  - Precedence preservation
  - Flexible configuration

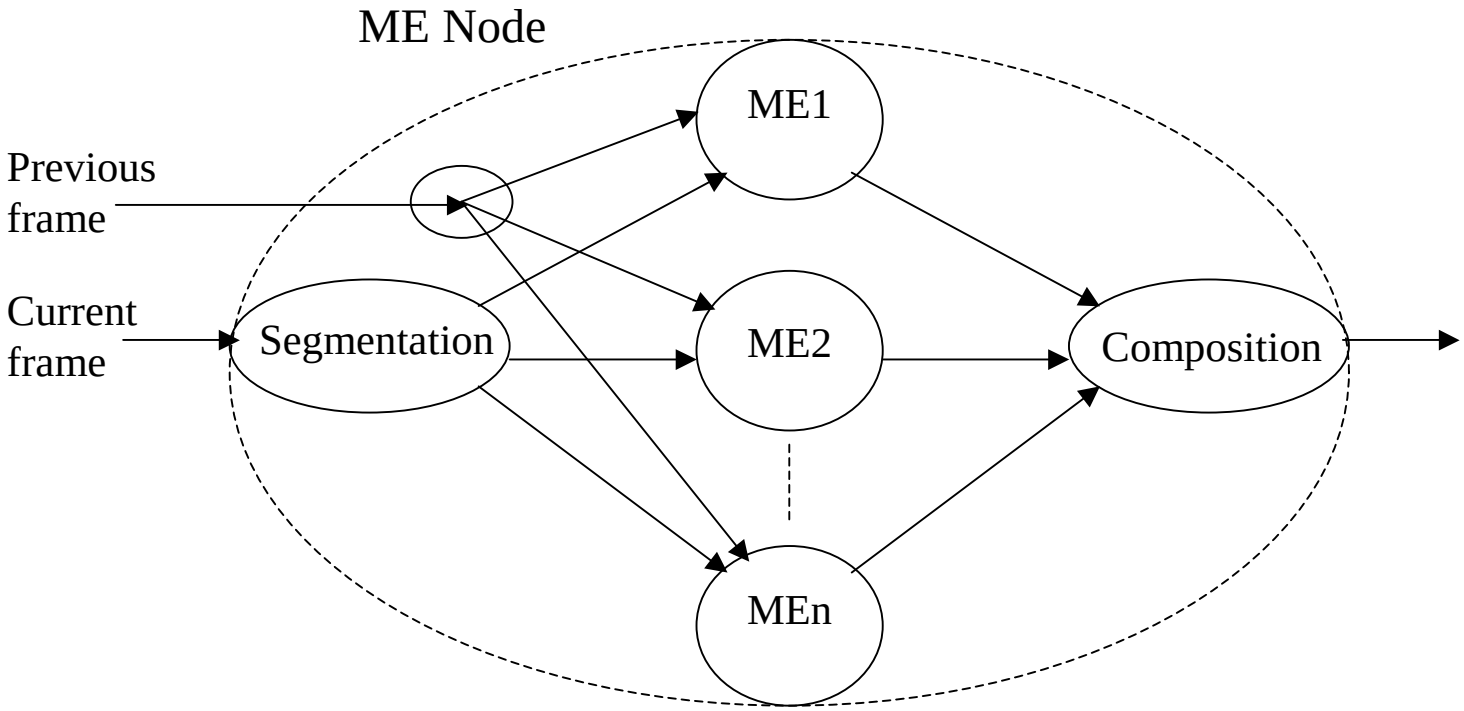
Our Approach

- System modeling using Computational Process Networks
  - Deterministic concurrent model
  - Precedence-preserving
- Software implementation
  - C++, POSIX Threads
  - Allen's CPN framework

PN Model of the Core Encoder



Finer Hierarchical Model of Motion Estimation Node



Software Implementation

- Node and queue design
  - Data type and structure for node input, node output and tokens
- Code generation (time-consuming!)
  - Based on existing C source code on the web
- Simulation
  - Frame-based top level core encoder
  - Platform: Single Intel Pentium III Xeon (733MHz?) processor, Linux, 256MB memory

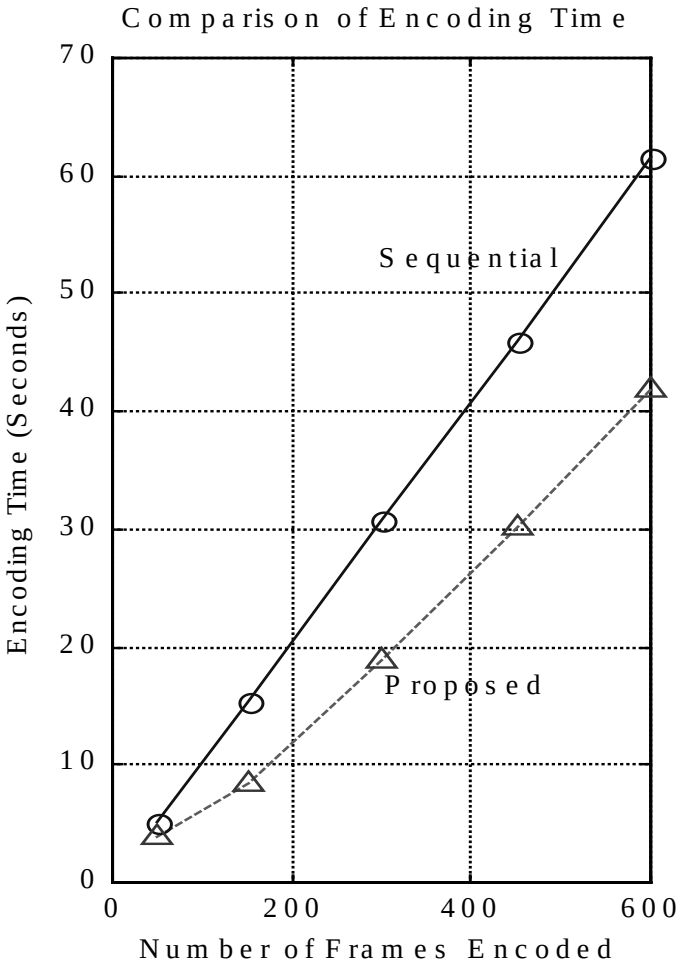
Example of Nodes Execution

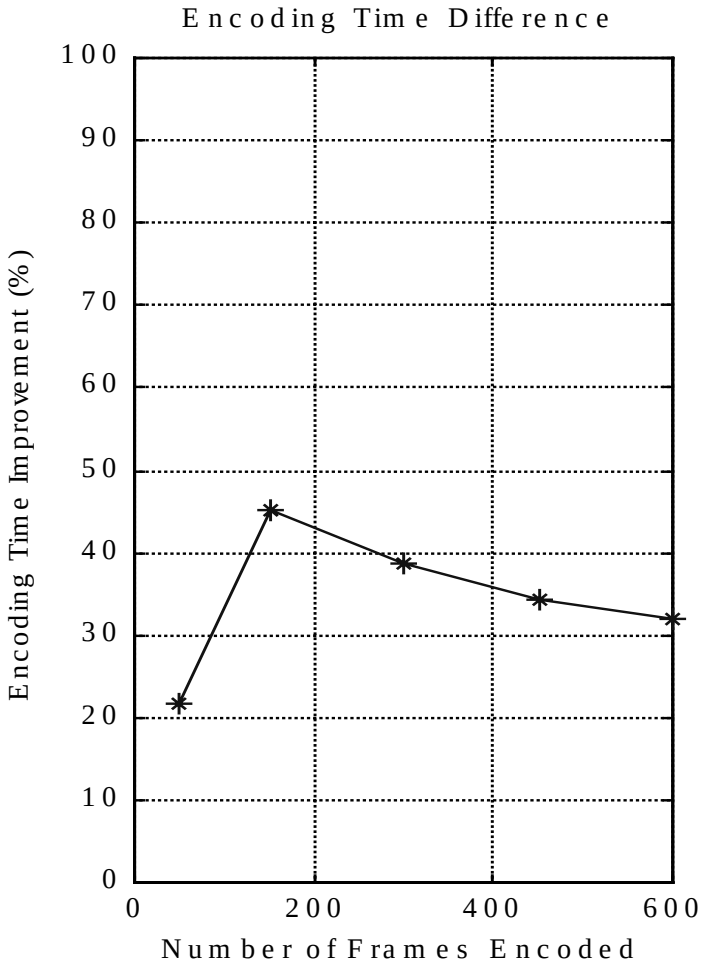
...
Encoding frame 0 ...
ForkNode starting .
ForkNode processed 1 frame(s).
ForkNode starting .
ForkNode processed 1 frame(s).
MENode starting.
MENode processed 1 frame(s).
MCNode starting.
MCNode processed 1 frame(s).
ForkNode starting .
ForkNode processed 1 frame(s).
SUBPredNode starting.
SUBPredNode processed 1 frame(s).
DCTNode starting.
DCTNode processed 1 frame(s).
QvlcNode starting.
QvlcNode processed 1 frame(s).
IQUANTNode starting.
IQUANTNode processed 1 frame(s).
IDCTNode starting.
IDCTNode processed 1 frame(s).
ADDPredNode starting .
ADDPredNode processed 1 frame(s).
...

Simulation Results

- Successful encoding results
  - On test sequences (128*128, color format 4:2:0)
  - Decodable and playable by existing MPEG player
- Faster than the original sequential encoder
  - Even on a single processor!
  - Benefits from concurrent model and Pthread implementation outweigh thread overheads
  - Benefit margin may depend on the inherent parallelism exposed by the designed model and node granularity

Performance Evaluation





Conclusion

- Our approach is
  - Scalable to multi-processor environment (expected to have approximate linear speedup thus potentially feasible for real-time implementation)
  - Faster due to concurrent execution (Pthread implementation of PN nodes)
- Future work
  - Profiling the computation load of each node
  - Evaluation on multi-processor platform