New Design of a MAP Decoder

Leila Sabeti

Advisor: Dr. M. Ahmadi
Co-Advisor: Dr. K. Tepe
April 2004
University Of Windsor
Outline

• Objective
• Digital Communication System
• Turbo Encoder
• Turbo Decoder
• Algorithms History
• Comparison
• BCJR/MAP Algorithm
• Max-Log-MAP Algorithm
• Comparison between previous implementations
• Proposed System Design
• Proposed quantization
• Metric normalization
• RTL simulation
• Synthesis
• Future works
Objective

• MAP/BCJR Decoder
  - can be used in communication systems (wireless, satellite, magnetic recording, digital video, …)
  - Minimizes the bit error rate of received channel information
  - Regenerates the original information
• Max-Log-MAP algorithm for implementation.
Digital communication System

1/2

RSC

BPSK

Information Source

Source Encoder

Channel Encoder

Digital Modulator

Transmitter

AWGN

Channel

Output information

Source Decoder

Channel Decoder

Digital Demodulator

Receiver
Turbo Encoder

- Recursive Systematic Convolutional Codes (RSCC), two memory, code rate 1/2.
- Parallel or Serial concatenation of (RSCC) and a pseudo random interleaver and/or more memories.
- The encoding process represented by a state transition diagram.

RSC Encoder (Two Memory, Rate ½, Generators (7,5))
Turbo Encoder

• Expanding the state transition diagram

![Trellis diagram for (7,5) convolutional code]
Turbo Decoder (SISO)

• Important development in coding theory in recent years.
• Standard (Consultative Committee for Space Data Systems (CCSDS), and 3rd Generation Partnership Project (3GPP))
• Strong requirement for the efficient implementation
Turbo Decoder

- MAP/BCJR Decoders, interleavers and deinterleavers
- BCJR algorithm for received channel sequences
- Passing information to the next decoder at each iteration
- Reduction of Bit Error Rate (BER).
Algorithms History

- 1967: Viterbi Algorithm (VA)[6]
- 1972: MAP/BCJR Algorithm[1]
- 1989: Optimum Update (SOVA-SU)[7]
- 1990: Max-Log-MAP[2]
- 2001: Improved Max-Log-MAP [4][5]
Performance of different Turbo decoders

- MAP and Log-MAP have the best accuracy.
- SOVA is the worst.
- ML-MAP is in between but it will be improved by iterative decoding and using scaling factor for APP.
## Complexity Comparison

<table>
<thead>
<tr>
<th>MAP/BCJR</th>
<th>Max-Log-MAP</th>
<th>Log-MAP</th>
<th>Sliding MAP</th>
<th>SOVA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O_M(n^2)$</td>
<td>$O_c(n^2)$</td>
<td>$O_c(n^2)$</td>
<td>$O_M(6n^2)$</td>
<td>$O_c(0.5n^2)$</td>
</tr>
<tr>
<td>$O_S(n^2)$</td>
<td>$O_S(n^2)$</td>
<td>$O_S(2n^2)$</td>
<td>$O_S(6n^2)$</td>
<td>$O_S(0.5n^2)$</td>
</tr>
</tbody>
</table>

- n: Number of states, M: Multiplications, S: Summations,
- C: Comparisons
- The differences of considered architecture in terms of power consumption is not significant.
- Improved ML-Map by using a scaling factor within the extrinsic calculation.
**BCJR/MAP Algorithm**

- The output of this algorithm (soft output) gives the probability of each received bit of information to be one or zero.

**State Trellis Length**

- **Forward Backward recursion**

Inputs

- **γ**
- **α**
- **β**

Soft Information

Output
BCJR/MAP Algorithm

\[ \gamma_t(m', m) = \sum_x p_t(m | m').q_t(X_t | m', m).R(Y_{td} | X_t).R(Y_{tp} | X_t) \]

\[ \alpha_t(m) = \sum_{m'} \alpha_{t-1}(m').\gamma_t(m', m) \]

\[ \beta_t(m') = \sum_m \beta_{t+1}(m).\gamma_{t+1}(m', m) \]

\[ \Lambda(X_{t+1}) = \ln \frac{\sum_{(m', m), X = 1} \gamma_{t+1}(m', m).\beta_{t+1}(m).\alpha_t(m')}{\sum_{(m', m), X = -1} \gamma_{t+1}(m', m).\beta_{t+1}(m).\alpha_t(m')} \]

• Too difficult in practice, because of the numerical representation of probabilities, nonlinear functions and mixed multiplications and additions of these values.
Max-Log-Map Algorithm

- work with the logarithms of the values using the following approximation:
  \[ \ln(e^{\gamma_1} + \ldots + e^{\gamma_n}) \approx \max_{i \in \{1, \ldots, n\}} \gamma_i \]

- Multipliers which make the design complex, huge and slow are changed to adders and comparators.

\[
\ln \gamma_t(m', m) = \frac{2Y_{t_d} X_t}{N_0} + \frac{2Y_{t_p} X_t}{N_0} + \ln AP_t + K
\]

\[
\ln \alpha_t(m) = \max_{m'}[\alpha_{t-1}(m') + \gamma_t(m', m)]
\]

\[
\ln \beta_t(m') = \max_{m}[\beta_{t+1}(m') + \gamma_{t+1}(m', m)]
\]
Max-Log-MAP Algorithm

- Using Alpha, Beta and Gamma, Log-Likelihood Ratio (LLR) is computed which provides soft decision.
- Soft Output makes it possible to decide if each received Bit of information is zero or one.

Log-Likelihood Ratio (LLR)

\[
\ln \Lambda_{t+1} = \max_{(m,m'),X=1} \left[ \ln \gamma_{t+1}(m',m) + \ln \beta_{t+1}(m) + \ln \alpha_t(m',m) \right] - \max_{(m,m'),X=-1} \left[ \ln \gamma_{t+1}(m',m) + \ln \beta_{t+1}(m) + \ln \alpha_t(m',m) \right]
\]
# Previous implementations

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Speed</th>
<th>Area</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4] ML-MAP</td>
<td>High</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>[12] Log-MAP</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>[8] SL-MAP</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Prp ML-MAP</td>
<td>High</td>
<td>↓</td>
<td>↑</td>
</tr>
</tbody>
</table>

- Speed range about 20MHz~100MHz, needed for iterations
- Minimum area about 7mm²

- Disadv. of [8]: Complex Control unit for synchronization of decoding steps
- Decreasing the memory size and increasing the accuracy in ML-MAP the lowest-complexity algorithm.
- Using the parallel calculation and LUTs, High speed
Proposed System Specification

- Encoder: Recursive Systematic Convolutional (RSC)
- Channel: Additive White Gaussian Noise (AWGN)
- Considered Modulation: Binary Phase Shift Keying (BPSK), which maps 1 to 1 and 0 to –1.
- Number Of Memories: 2.
- Code Rate: R=1/2
- Block size: Flexible to the block size
Proposed System Design

1. Gamma and Alpha are calculated together and stored in RAM.
2. Beta and Landau are also calculated in parallel to give the soft output.
3. Faster, less memory and reduced area.
Proposed Gamma Unit

- Logarithm of Gamma

\[
\ln \gamma_t(m', m) = \frac{2Y_{td} X_t}{N_0} + \frac{2Y_{tp} X_t}{N_0} + \ln AP_t + K
\]

- No sensitivity of Max-Log-MAP algorithm to the variance of the noise
- Eight nonzero Gammas but four different values.

\[
\begin{align*}
\ln \gamma_{t,00}(m', m) &= (-Y_{td} - Y_{tp}) + \ln AP_t (-1) \\
\ln \gamma_{t,01}(m', m) &= (-Y_{td} + Y_{tp}) + \ln AP_t (-1) \\
\ln \gamma_{t,10}(m', m) &= (+Y_{td} - Y_{tp}) + \ln AP_t (+1) \\
\ln \gamma_{t,11}(m', m) &= (+Y_{td} + Y_{tp}) + \ln AP_t (+1)
\end{align*}
\]
Yd (systematic data) and Ys (Parity data) are added/subtracted.
Alpha Calculation Unit

• In each Block Alpha is calculated using proper Gamma and previous calculated Alphas.
• Beta Calculation Unit
• Soft outputs
Proposed quantization

- Quantization of input, Gamma, Alpha, Beta, Output and...
- Decreasing the number of bits -> Lower accuracy
- Increasing -> Larger memories for storage
- Crucial choosing
Proposed quantization

- Decoder inputs [-4, 4], 90% covering.
- Integer value with one digit precision.

- APP values between –8 and +8.
Proposed quantization

- $\ln \text{AP}(1)$ and $\ln \text{AP}(-1)$ are quantized to integer values from –8 to 0.
- Also 8bits for $\gamma$, $\alpha$, $\beta$ and 8bits for output is considered.
Metric Normalization

- In forward or backward recursions, metric values can easily overflow or underflow.
- Subtraction of the maximum or minimum node metrics at a specific time from all of the node metrics at that time
RTL Simulation

- Verilog
- Simvision
• Synopsys (Design analyzer)
• Modules
• Synopsys
• I/O wrapper
Synthesis

- Synopsys
- Area: 0.96 mm\(^2\)
- Speed: 150 MHz
- Fastest Implementation: 110 MHz
Partitioning and Floorplanning

• Encounter
Future Works

System Design

RTL Simulation

Synthesis

Scan Insertion

Gate-Level Simulation

Floorplanning

Placement

Clock Tree Generation

Routing & Timing Verification

Physical Verification
Thank you