# A New Computationally Efficient Bit-Loading Algorithm for the Margin Maximization Problem in DMT Systems

*N. Papandreou and Th. Antonakopoulos*

University of Patras, Greece

Dept. of Electrical Engineering and Computers Technology

*Communications and Embedded Systems Group*

e-mail: {npapandr, theodore}@loe.ee.upatras.gr

# Outline

- Problem Formulation
- New Approach
  - *Initial Bit-Allocation*
  - *Intermediate and Final Bit-Allocation*
- Numerical Results
- Performance  Improvement
- Conclusions

# Problem Formulation

*…allocate bits and power over the subchannels*

$$\text{minimize}: \quad P_{total} = \sum_{i=1}^{N} P_i$$

$$\text{subject to}: \quad \sum_{i=1}^{N} b_i = B_{target}$$

$$0 \le P_i \le P_{max}$$

$$0 \le b_i \le b_{max}$$

$$P_i^{error} \le P_{max}^{error}$$

$$b_i \in Z_+$$

## Optimum Discrete Solutions

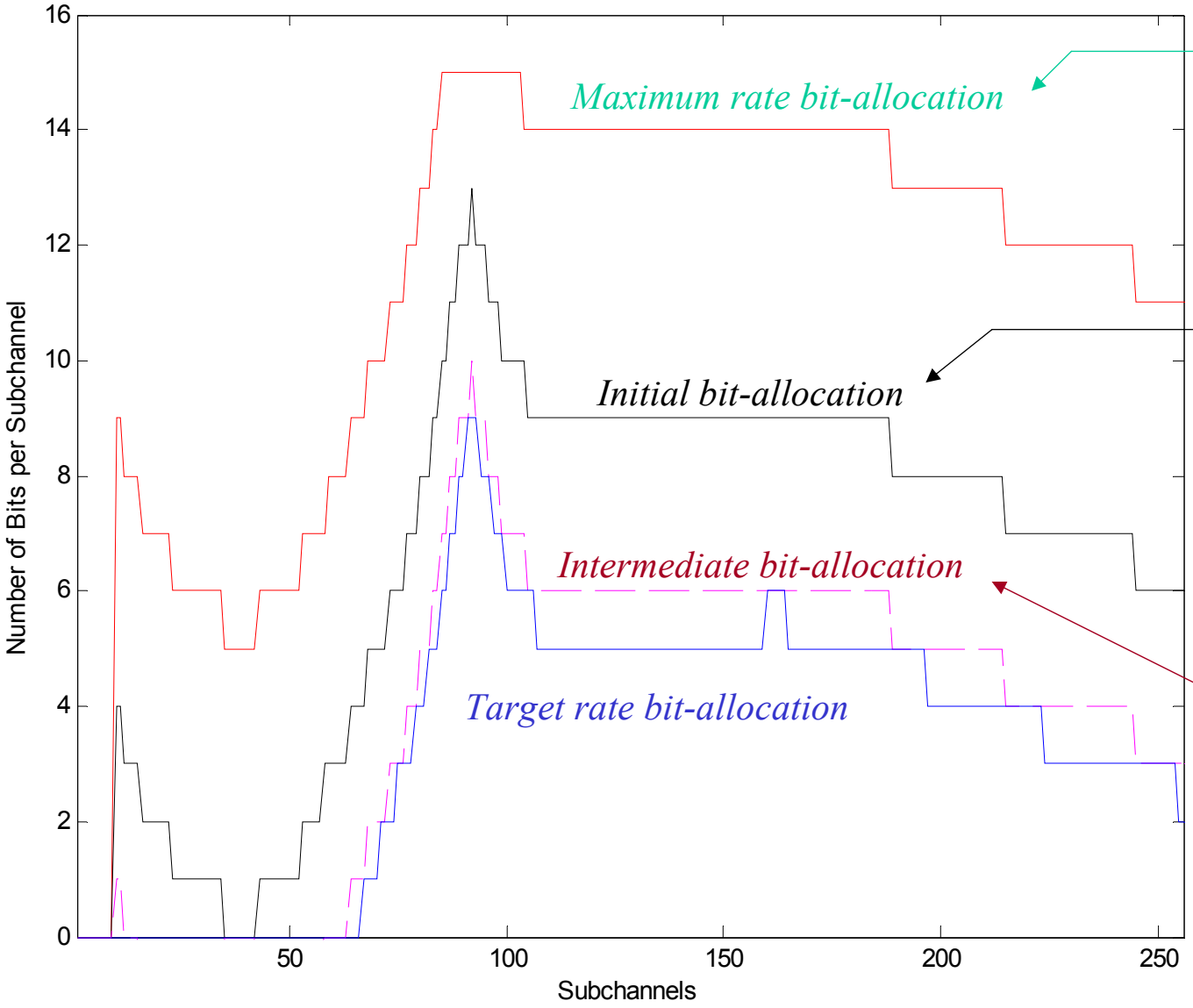
- Greedy Bit-Filling or Bit-Removal (*Levin-Campello*)

## Sub-Optimum Non-Discrete Solutions

- Direct or Look-Up Table methods based on Langrange Multipliers (*Waterfilling*)

*system constraints*

# New Approach




…rate upper bound: depends on bit and power constraints

…loop characteristic bit-profile: depends on subchannel gain-to-noise ratio values

*multiple bits insertion or removal*

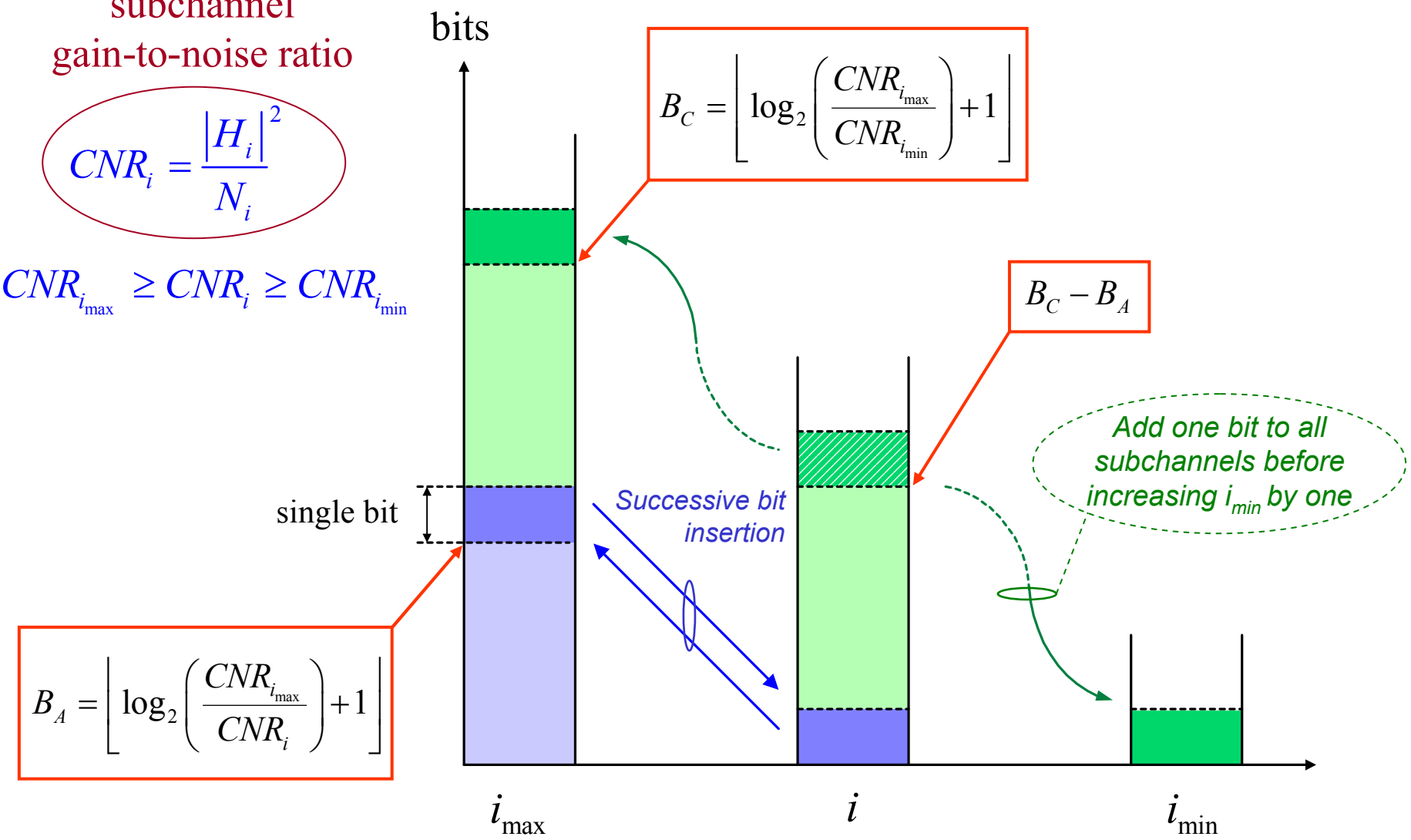
…approach target rate with no more than a single bit difference per subchannel

# Initial Bit-Allocation (1)

subchannel gain-to-noise ratio

$$CNR_i = \frac{|H_i|^2}{N_i}$$

$$CNR_{i_{\max}} \geq CNR_i \geq CNR_{i_{\min}}$$



# Initial Bit-Allocation (2)

*…loop's characteristic bit profile:*

$$b_i' = \begin{cases} \left\lfloor \log_2(k_{i_{\min}}) + 1 \right\rfloor, & i = i_{\max} \\ \left\lfloor \log_2(k_{i_{\min}}) \right\rfloor - \left\lfloor \log_2(k_i) \right\rfloor, & i \neq i_{\max} \end{cases}$$

$$k_i = \frac{CNR_{i_{\max}}}{CNR_i}, \ \forall i$$

*might be invalid under bit and power constraints*

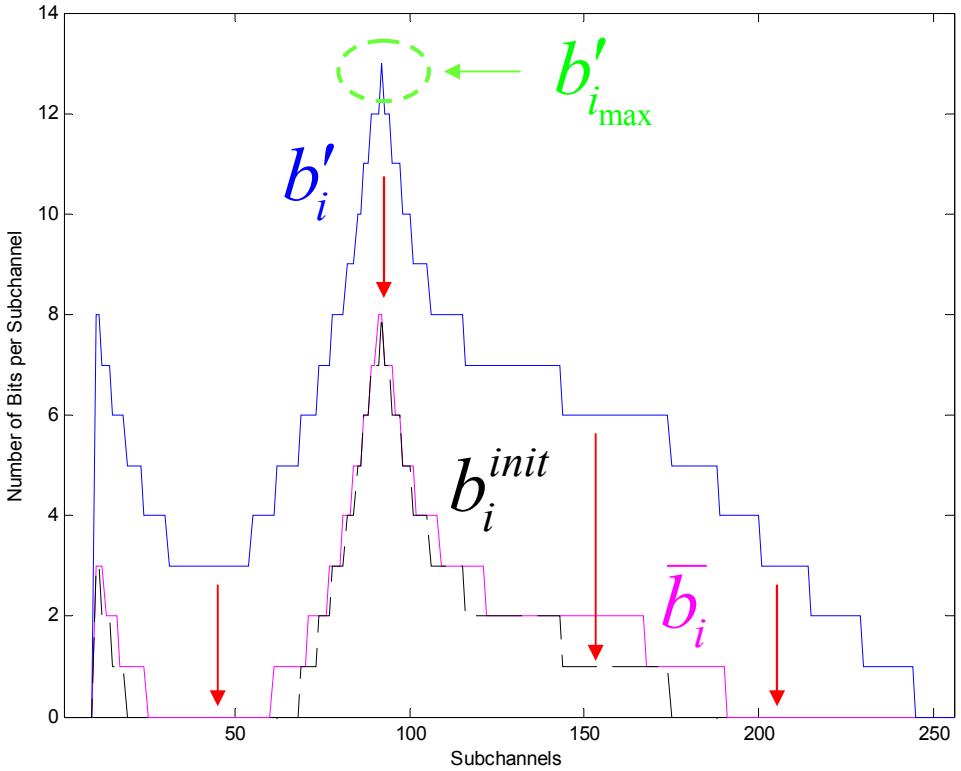
*…loop's bit upper-bound:*

$$\bar{b}_i = \min\left\{ b_{\max}, \left\lfloor \log_2\left(1 + \frac{\bar{P}_i \cdot CNR_i}{\Gamma}\right) \right\rfloor \right\}$$

*…initial bit profile:*

down shift

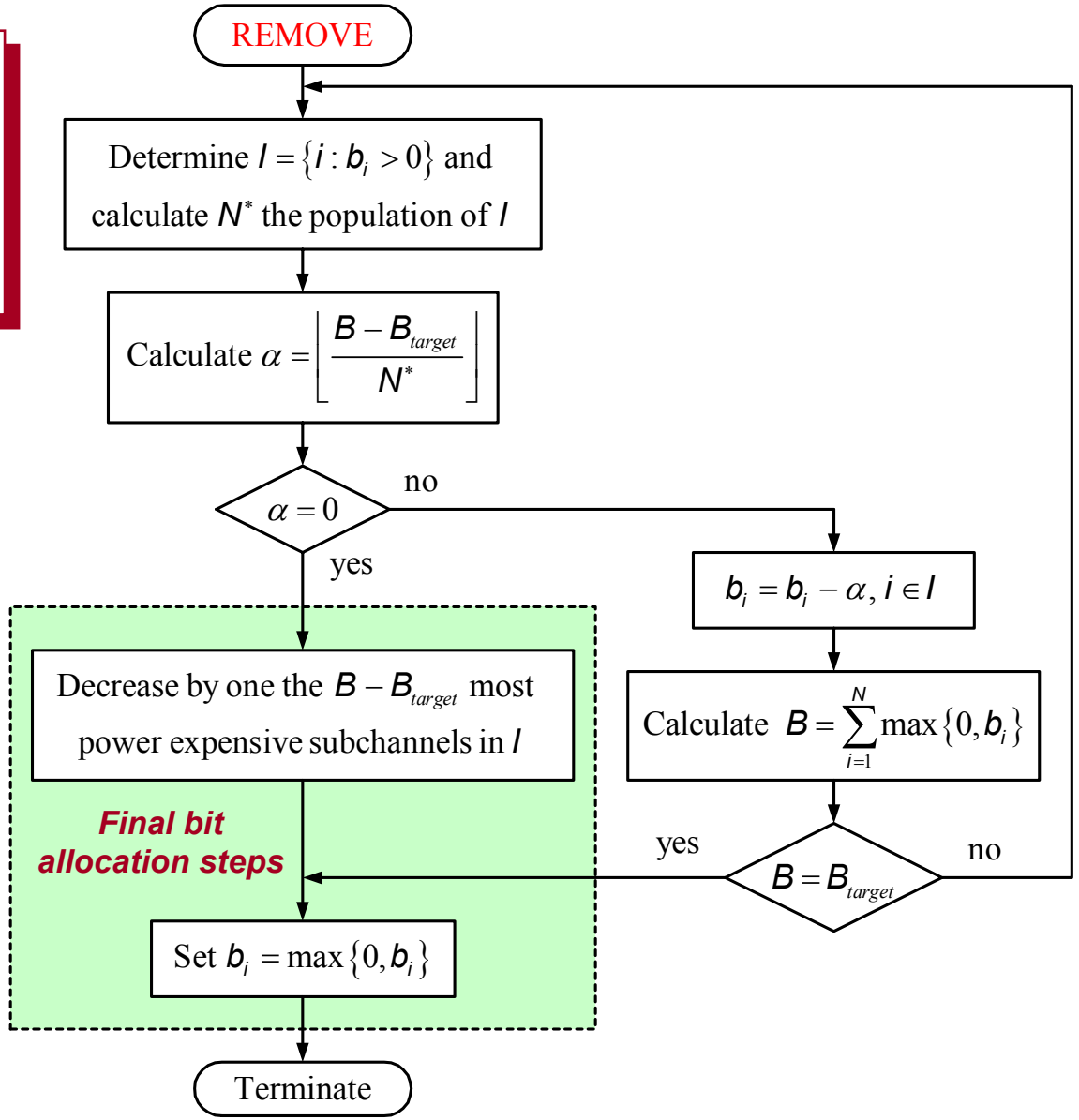
$$b_i^{init} = \begin{cases} b_i' - \left(b_{i_{\max}}' - \bar{b}_{i_{\max}}\right), & \text{if } b_{i_{\max}}' > \bar{b}_{i_{\max}} \\ b_i', & \text{otherwise} \end{cases}$$



# Intermediate & Final Bit-Allocation

**Multiple bits removal:**

*…approach target rate with no more than a single bit difference per subchannel*

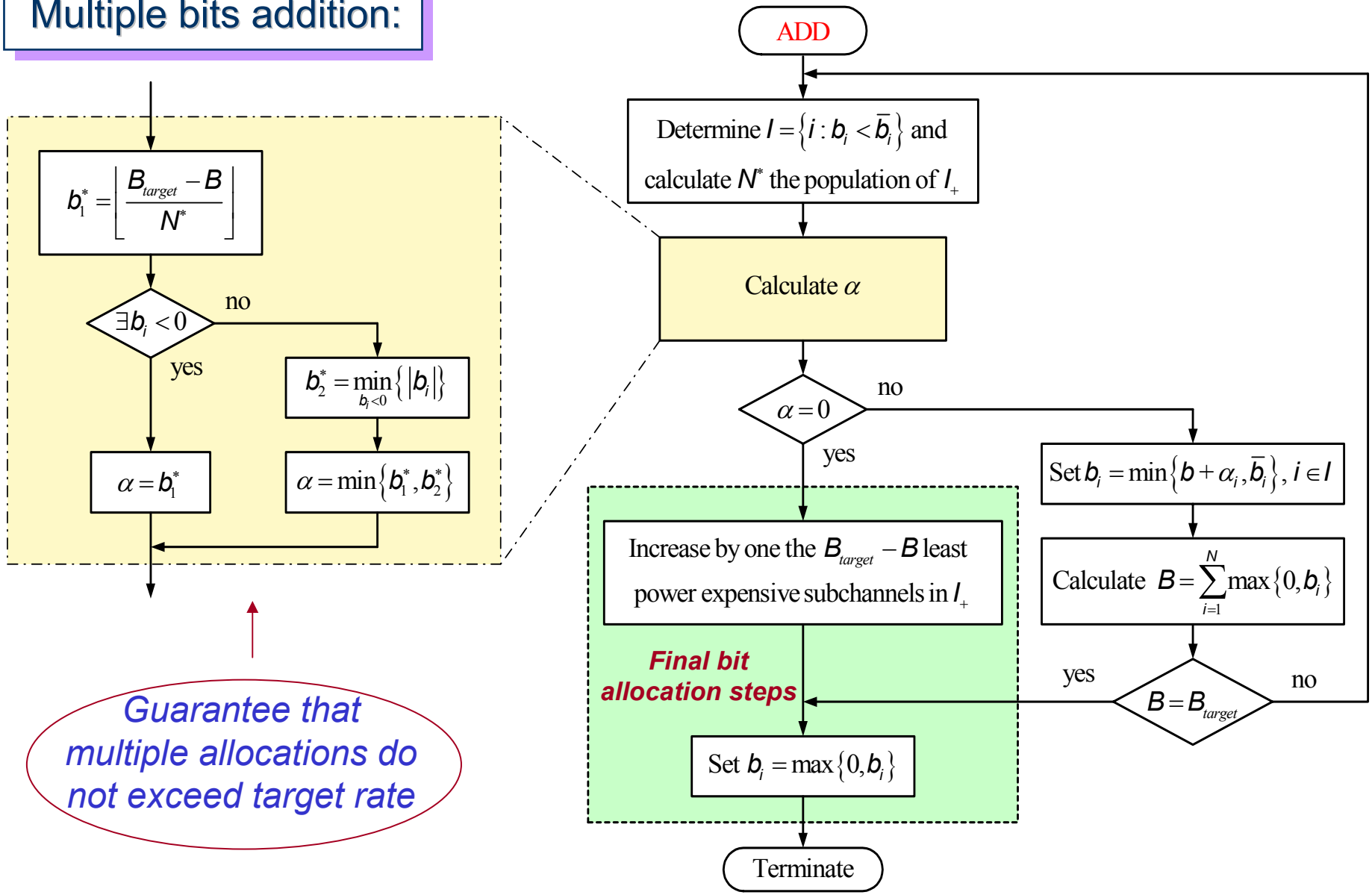



*Greedy bit-removal for at most N bits*

*Most computationally expensive part of the algorithm*

# Intermediate & Final Bit-Allocation

## Multiple bits addition:



# ADSL Test-loops

| Test loops | Margin (dB) | Crosstalk Disturbers | | | |
|---|---|---|---|---|---|
| | | ADSL FEXT | HDSL NEXT | ISDN NEXT | T1 (adj.) NEXT |
| T1.601 #7 | 6 | - | - | 24 | - |
| T1.601 #13 | 6 | - | - | 24 | - |
| CSA #4 | 6 | 24 | - | 24 | - |
| CSA #6 | 6 | - | 20 | - | - |
| CSA #7 | 6 | 10 | - | 10 | - |
| mid-CSA | 3 | - | - | 24 | 10 |

ANSI T1.417-1995

# Numerical Results

| ANSI T1.413-1995 ADSL test loops | Target rate (bits/symbol) | Algorithm's bit-allocation phases | | |
|---|---|---|---|---|
| | | *Initial* | *Intermediate* | *Final* |
| | | Total rate (bits/symbol) | Multiple allocations | Remaining bits |
| CSA #6 | 10% 151 | 992 | 3 | 6 |
| | 50% 754 | | 1 | 16 |
| | 90% 1357 | | 1 | 118 |

**of loop's maximum rate**

| ANSI T1.413-1995 ADSL test loops | Target rate (bits/symbol) | Execution time (msec) | | | Improvement factor | |
|---|---|---|---|---|---|---|
| | | *bit-filling* | *bit-removal* | NEW algorithm | *bit-filling* | *bit-removal* |
| CSA #6 | 151 | 24.84 | 132.82 | 18.07 | 1.37 | 7.35 |
| | 754 | 80.12 | 82.88 | 20.47 | 3.91 | 4.05 |
| | 1357 | 133.24 | 29.63 | 28.60 | 4.66 | 1.04 |

**Loop: 9 kft, 26 AWG**

**Noise: 20 HDSL NEXT+AWGN**

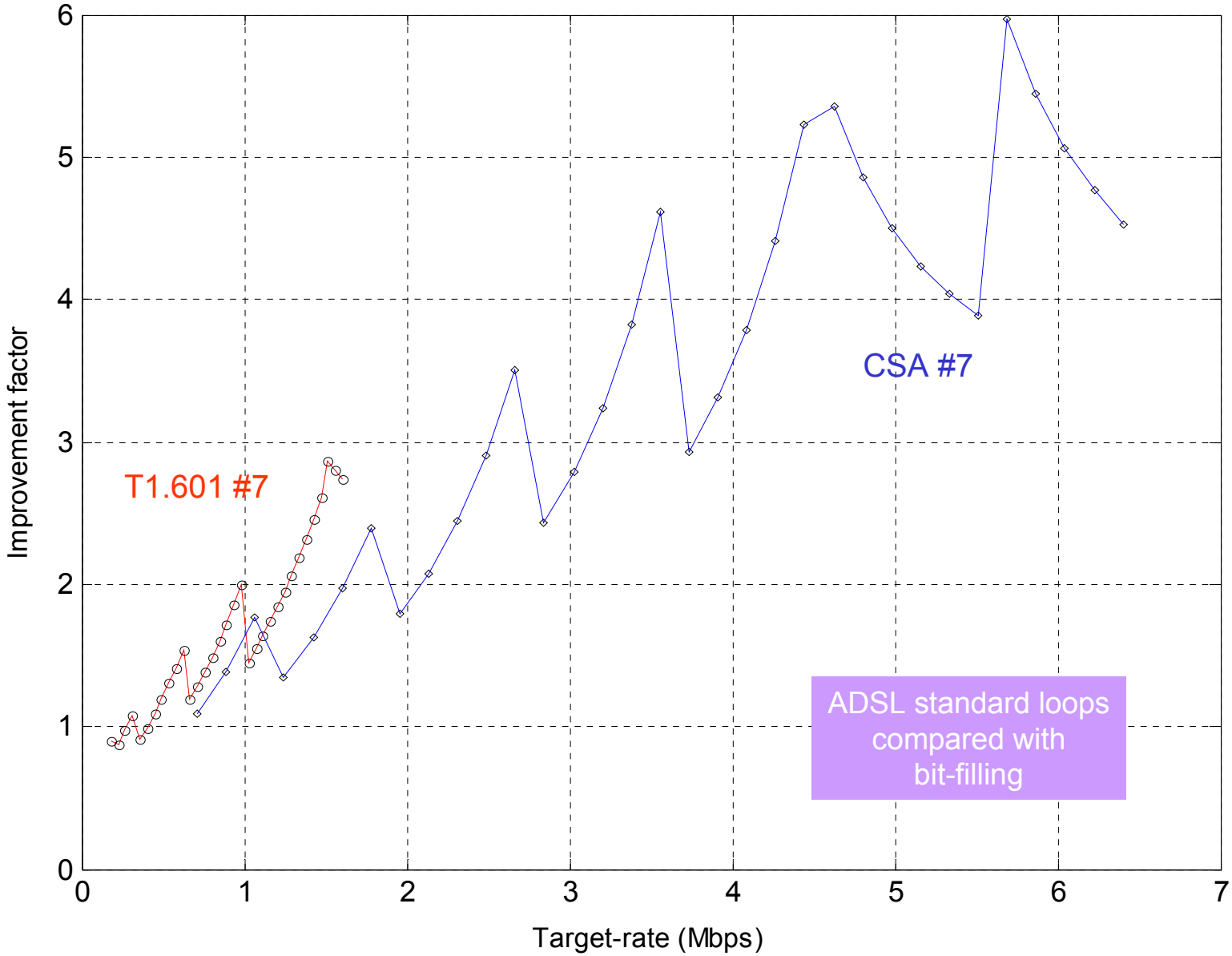
# Numerical Results

| ANSI T1.413-1995 ADSL test loops | Target rate (bits/symbol) | Algorithm's bit-allocation phases | | |
|---|---|---|---|---|
| | | *Initial* | *Intermediate* | *Final* |
| | | Total rate (bits/symbol) | Multiple allocations | Remaining bits |
| T1.601 #7 | 10% 45 | 374 | 3 | 5 |
| | 50% 223 | | 1 | 31 |
| | 90% 401 | | 0 | 27 |
| T1.601 #13 | 59 | 526 | 2 | 28 |
| | 297 | | 1 | 83 |
| | 534 | | 0 | 8 |
| CSA #4 | 171 | 786 | 1 | 65 |
| | 856 | | 0 | 70 |
| | 1540 | | 1 | 13 |
| CSA #6 | 151 | 992 | 3 | 6 |
| | 754 | | 1 | 16 |
| | 1357 | | 1 | 118 |
| CSA #7 | 178 | 1132 | 2 | 118 |
| | 889 | | 1 | 4 |
| | 1600 | | 1 | 221 |
| mid-CSA | 185 | 601 | 2 | 38 |
| | 924 | | 2 | 96 |
| | 1662 | | 2 | 7 |

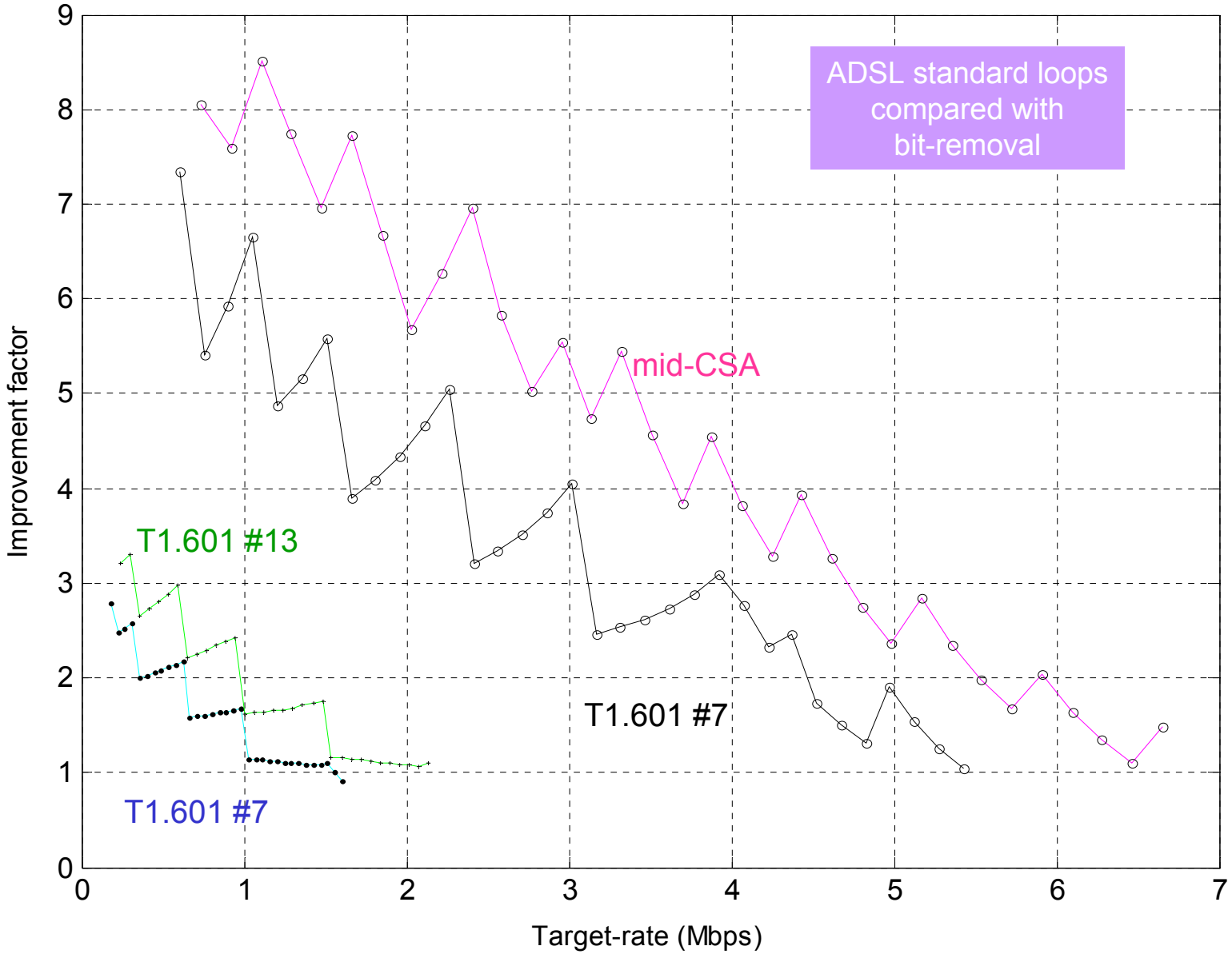
# Numerical Results

| ANSI T1.413-1995 ADSL test loops | Target rate (bits/symbol) | Execution time (msec) | | | Improvement factor | |
|---|---|---|---|---|---|---|
| | | *bit-filling* | *bit-removal* | NEW algorithm | *bit-filling* | *bit-removal* |
| T1.601 #7 | 10% 45 | 12.75 | 39.76 | 14.24 | 0.90 | 2.80 |
| | 50% 223 | 29.05 | 27.67 | 16.85 | 1.72 | 1.64 |
| | 90% 401 | 43.14 | 14.33 | 15.72 | 2.74 | 0.91 |
| T1.601 #13 | 59 | 14.58 | 51.84 | 16.17 | 0.90 | 3.20 |
| | 297 | 36.34 | 35.13 | 21.28 | 1.71 | 1.65 |
| | 534 | 55.53 | 16.63 | 15.20 | 3.65 | 1.09 |
| CSA #4 | 171 | 25.60 | 151.78 | 29.10 | 0.91 | 5.21 |
| | 856 | 89.41 | 94.54 | 25.11 | 3.56 | 3.76 |
| | 1540 | 150.69 | 31.50 | 20.30 | 7.42 | 1.55 |
| CSA #6 | 151 | 24.84 | 132.82 | 18.07 | 1.37 | 7.35 |
| | 754 | 80.12 | 82.88 | 20.47 | 3.91 | 4.05 |
| | 1357 | 133.24 | 29.63 | 28.60 | 4.66 | 1.04 |
| CSA #7 | 178 | 27.27 | 154.48 | 25.08 | 1.08 | 6.16 |
| | 889 | 92.55 | 98.94 | 20.06 | 4.61 | 4.93 |
| | 1600 | 156.53 | 32.27 | 34.63 | 4.52 | 0.93 |
| mid-CSA | 185 | 27.94 | 151.09 | 18.75 | 1.50 | 8.06 |
| | 924 | 95.83 | 95.72 | 24.96 | 3.84 | 3.84 |
| | 1662 | 165.10 | 33.06 | 22.22 | 7.43 | 1.49 |

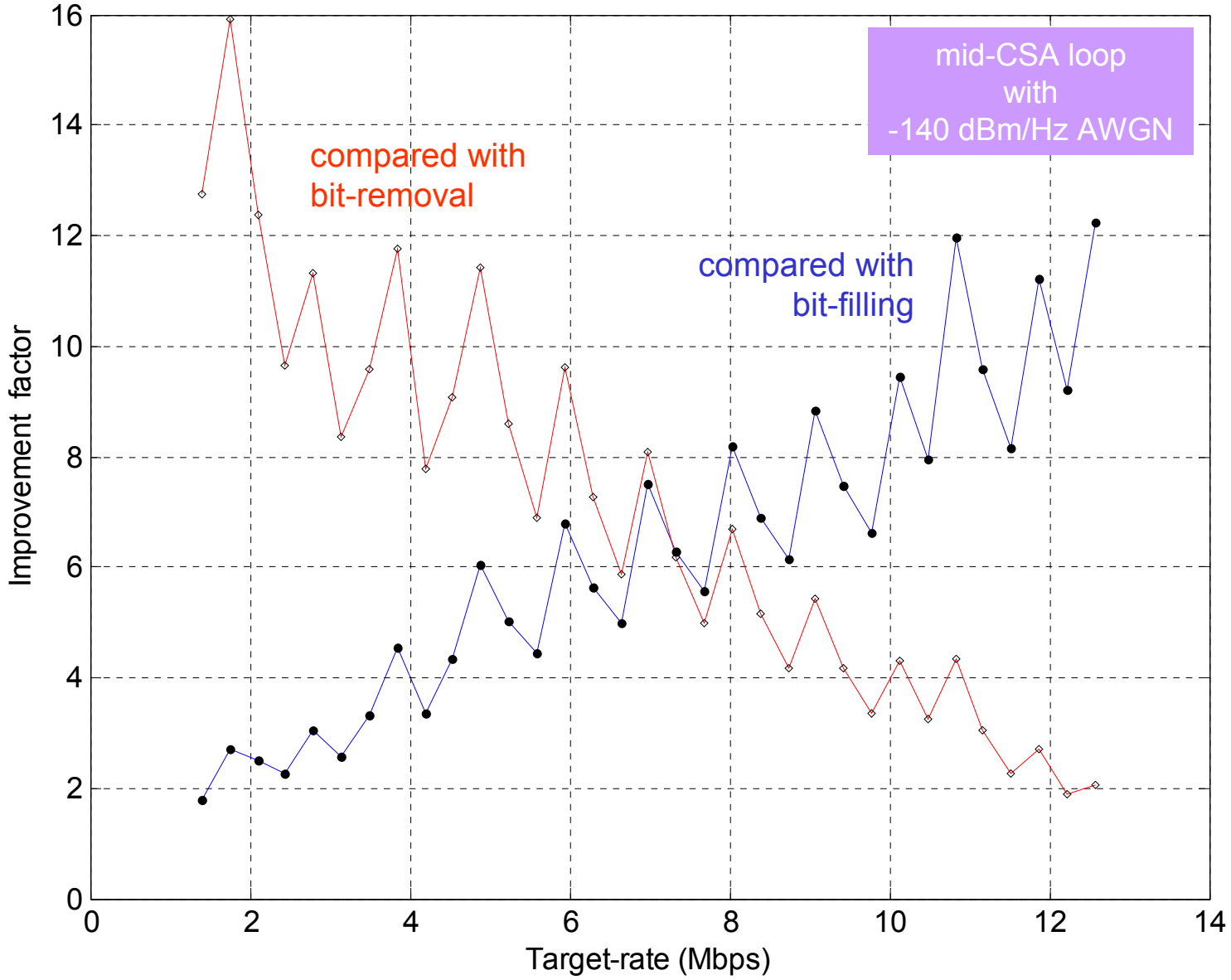
# Performance Improvement (1)



# Performance Improvement (2)



# Performance Improvement (3)



# Conclusions

*A new computationally efficient bit-loading algorithm for the margin maximization problem in DMT systems was presented.*

***Main features:***

- Exploit the loop's characteristic bit-profile based on the subchannel gain-to-noise ratio values
- Straightforward calculation of initial bit-allocation
- Multiple bits removal or addition process that converges fast to the final optimum bit distribution
- Greedy bit-filling or bit-removal steps for at most *N* bits