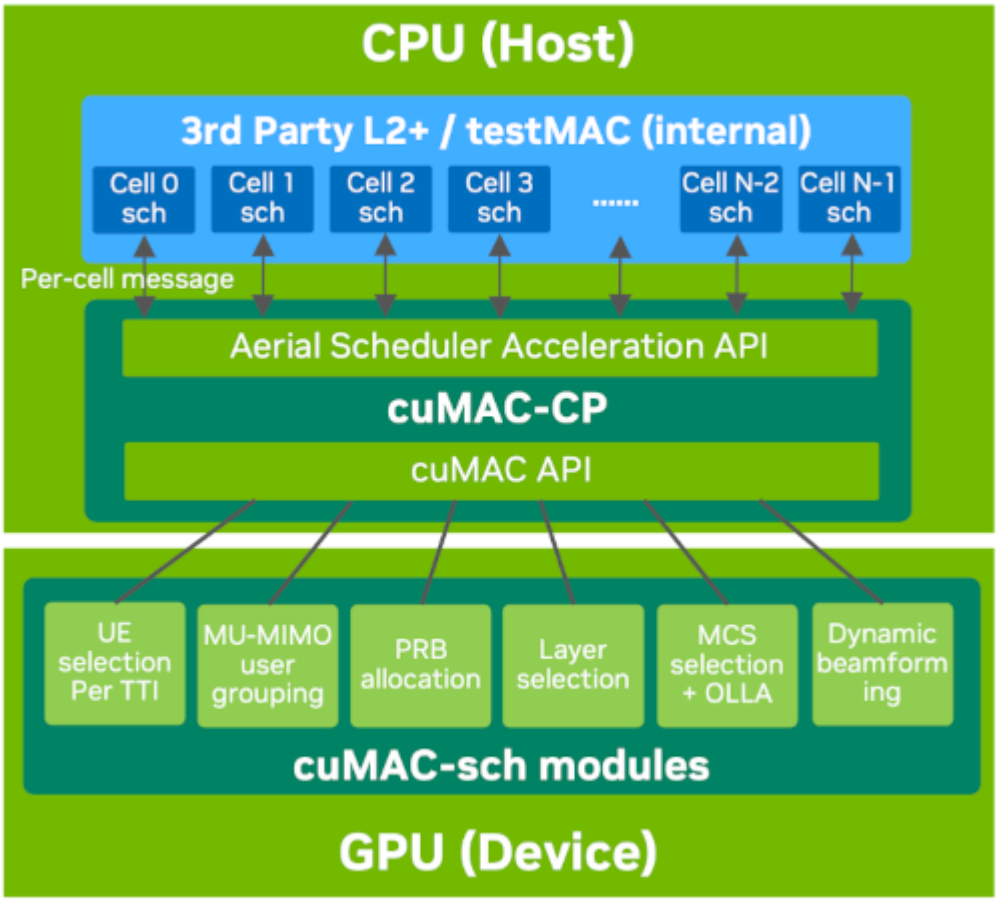
Aerial cuMAC

Aerial cuMAC is a CUDA-based platform for accelerating 5G/6G MAC layer scheduler functions with NVIDIA GPUs. cuMAC supported scheduler functions include UE selection/grouping, PRB allocation, layer selection, MCS selection/link adaptation and dynamic beamforming, all designed for the joint scheduling of multiple coordinated cells. cuMAC offers a C/C++ based API for the offloading of scheduler functions from the L2 stack in the DUs to GPUs. In the future, cuMAC will evolve into a platform that combines AI/ML based scheduler enhancements with GPU acceleration.



Aerial L2 scheduler acceleration data flow chart

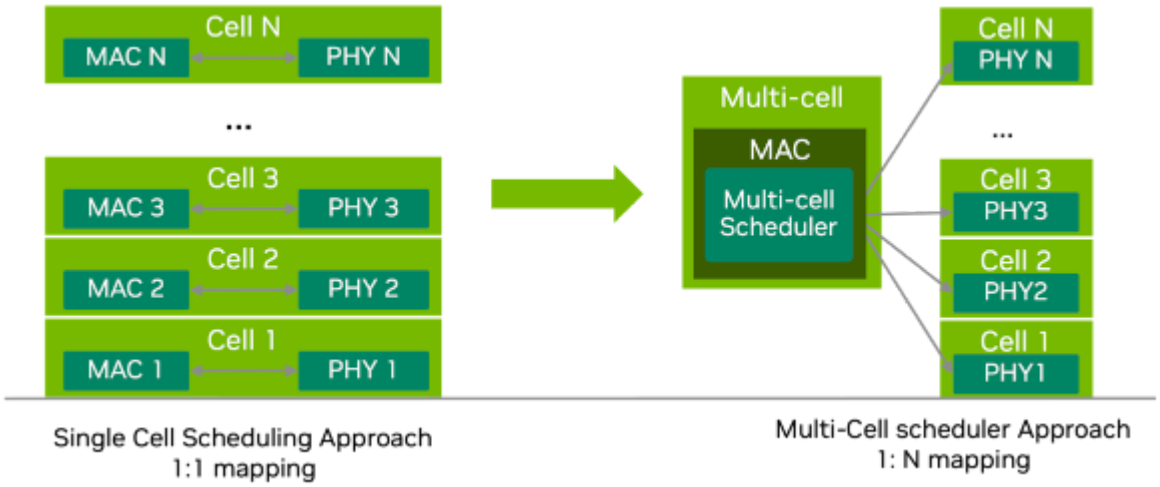
cuMAC is the main component of the Aerial L2 scheduler acceleration solution. The figure above illustrates the overall data flow of the scheduler acceleration. The full solution consists of the following components: 1) Aerial Scheduler Acceleration API, which is a per-cell message passing-based interface between the 3rd party L2 stack on DU/CU and cuMAC-CP, 2) cuMAC-CP, 3) cell group-based cuMAC API, and 4)cuMAC multi-cell scheduler (cuMAC-sch) modules.

The 3rd party L2 stack sits on the CPU and contains a single-cell L2 scheduler for each individual cell under its control. To offload L2 scheduling to GPU for acceleration/performance purposes, in each time slot (TTI), the L2 stack host sends per-cell request messages to cuMAC-CP through the Aerial Scheduler Acceleration API, which consists of required scheduling input & config. information from each single-cell scheduler. Upon receiving the per-cell request messages, cuMAC-CP integrates all scheduler input information from those (coordinated) cells into the cuMAC API cell group data structures and populates the GPU data buffers contained in these structures. Next, the cuMAC multi-cell scheduler (cuMAC-sch) modules are called by cuMAC-CP through cuMAC API to compute scheduling solutions for the given time slot (TTI). After the cuMAC-sch modules complete the computation and the scheduling solutions become available in the GPU memory, cuMAC-CP converts them into per-cell response messages and sends them back to the L2 stack host on CPU through the Aerial Scheduler Acceleration API. Finally, the L2 stack host uses the obtained solutions to schedule the cells under its control.

When there are multiple coordinated cell groups, a separate set of Aerial Scheduler Acceleration API, cuMAC-CP, cuMAC API and cuMAC instances should be constructed and maintained for each cell group.

Implementation Details

- **Multi-cell scheduling -** All cuMAC scheduling algorithms are implemented as CUDA kernels that are executed by GPU and jointly compute the scheduling solutions (PRB allocation, MCS selection, layer selection, etc.) for a group of cells at the same time. The algorithms can be constrained to single cell scheduling by configuring a single cell in the cell group. A comparison between the single-cell scheduler and multi-cell scheduler approaches is given in the below figure.



Single-cell scheduler approach vs. multi-cell scheduler approach

- **Scheduling algorithm CUDA implementation**

    - **PF UE down-selection algorithm -** cuMAC offers a PF-based UE selection algorithm to down-select a subset of UEs for new transmissions or HARQ re-transmissions in each TTI from the pool of all active UEs in each cell of a cell group. The association of UEs and cells in the cell group is an input to the UE selection module. When selecting UEs for each cell in each TTI, the UE selection algorithm first assigns a priority weight to each active UE in a cell and then sorts all active UEs in descending order of the priority weight. The subset of UEs that have the highest priority weights in each cell are selected for scheduling in a TTI. The number of selected UEs per cell is an input parameter to this module. HARQ re-transmissions are always assigned with the highest priority weight. For the new-transmission UEs, their priority weights are the PF metrics, calculated as the ratio of each UE's long-term average throughput and its instantaneous achievable data rate. The UE selection algorithm is implemented as CUDA kernels that run on GPU and jointly select UEs for all cells in a cell group at the same time.

    - **PF PRB allocation algorithms -** cuMAC offers algorithms to perform channel-aware and frequency-selective PRB allocation among a group of cells and their connected active UEs on a per-TTI basis. The input arguments to the PRB allocation algorithms include the narrow-band SRS channel estimates (MIMO channel matrices) per cell-UE link, the association solutions between cells and UEs, and other UE status and cell group parameters. The output is the PRB allocation solution for the cell group, whose data format depends on the type of allocation: 1) for type-0 allocation, a per UE binary bitmap indicating whether each PRB is allocated to the UE, and 2) for type-1 allocation, with 2 elements per UE indicating the starting and ending PRB indices for the UE's allocation. Two versions of the PRB allocation algorithms are provided, one for single cell scheduling and the other for multi-cell joint scheduling. A major difference between the two versions is that the multi-cell algorithm considers the impact of inter-cell interference in the evaluation of per-PRB SINRs, which can be derived from the narrow-band SRS channel estimates. The single-cell version does not explicitly consider inter-cell interference and only utilizes information restricted to each individual cell. The multi-cell algorithm can lead to a globally optimized resource allocation in a cell group by leveraging all available information from the coordinated multiple cells. A prototyping CUDA kernel implementation of PRB allocation algorithms is provided in the figure below.

- **Layer selection algorithm -** cuMAC offers layer selection algorithms that choose the best set of layers for transmission for a UE based on the singular value distribution across the UE's multiple layers. A predetermined singular value threshold is used to find the number of layers (with descending singular values) that can be supported on each subband (PRB group). Then the minimum number of layers across all allocated subbands to the UE is chosen as the optimal layer selection solution. Input arguments to the layer selection algorithms include the PRB allocation solution per UE, the singular values of each UE's channel on its allocated subbands, the association solutions between cells and UEs, and other UE status and cell group parameters. The output is the per-UE layer selection solution. The layer selection algorithm is implemented as CUDA kernels that run on GPU and jointly select layers for all UEs in a cell group at the same time.

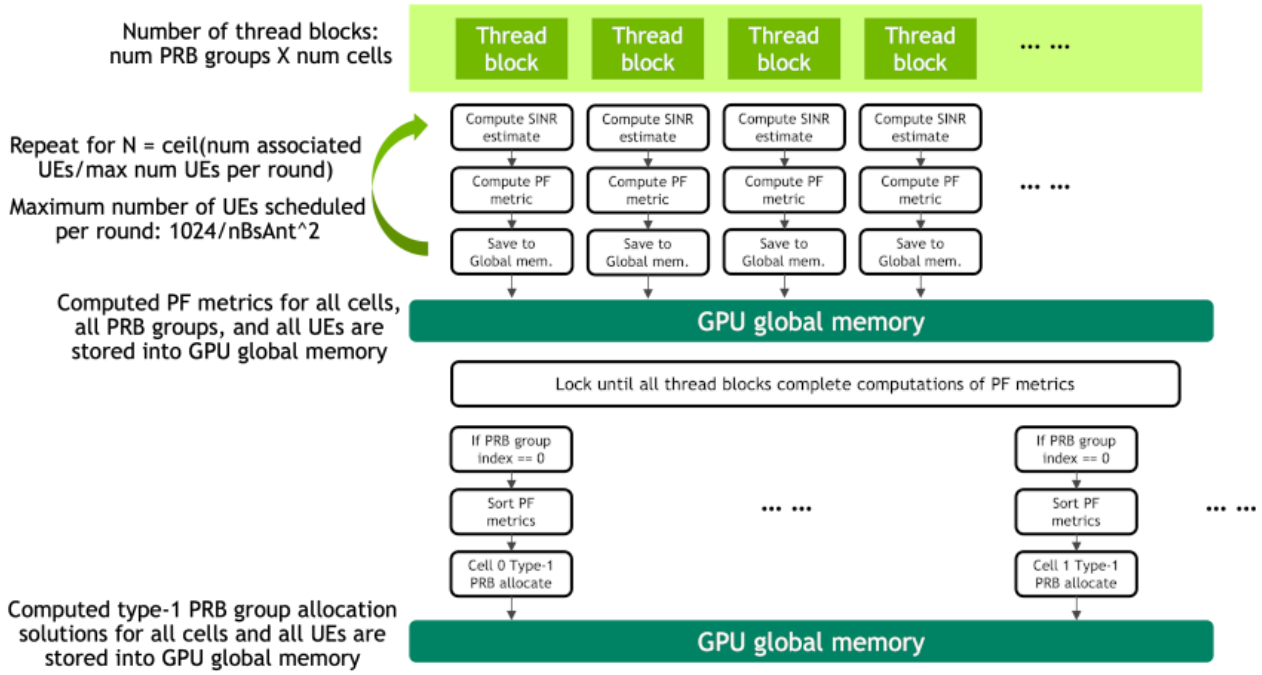
- **MCS selection algorithm -** cuMAC offers MCS selection algorithms that choose the best feasible MCS (highest level that can meet a given BLER target) per UE based on a given PRB allocation solution. An outer-loop link adaptation algorithm is integrated internally to the MCS selection algorithm, which offsets the SINR estimates based on previous transport block decoding results per UE link. Input arguments to the MCS selection algorithms include the PRB allocation solution per UE, the narrow-band SRS channel estimates (MIMO channel matrices) per cell-UE link, the association solutions between cells and UEs, the decoding results of the last transport block for each UE, and other UE status and cell group parameters. The output is the per-UE MCS selection solution. The MCS selection algorithm is implemented as CUDA kernels that run on GPU and jointly select MCS for all UEs in a cell group at the same time.

- **Support for HARQ -** all the above cuMAC scheduler algorithms can support HARQ re-transmissions with non-adaptative mode, i.e., reusing the same scheduling solution of the initial transmission for re-transmissions.

- **CPU reference code -** CPU C++ implementation of the above algorithms is also provided for verification and performance evaluation purposes.

- **Different CSI types -** cuMAC offers scheduler algorithm CUDA kernels to work with different CSI types, including SRS channel coefficient estimates and CSI-RS based channel quality information.

- **Support for FP32 and FP16 -** cuMAC offers scheduler algorithm CUDA kernels implemented in FP32 and FP16. Using FP16 kernels can help reduce scheduler latency with a minor performance loss.



A prototyping CUDA kernel implementation of PRB allocation algorithms

© Copyright 2024, NVIDIA.. PDF Generated on 06/06/2024