**针对异构内存系统的访存模式检测与优化工具**

**项目背景**：如今NVM、HBM等新型内存架构已经趋向成熟，更加复杂的包含异构内存系统（heterogeneous memory systems, HMS）的计算系统已经出现在商业市场当中，例如DRAM+MCDRAM的Intel KNL平台，Intel Opaque NVM+DRAM混合内存的计算平台等。此外，在广泛使用的GPU计算单元中，也存在多种存储单元（global, share, texture, constant memory）。而每种内存单元各有千秋，例如NVM具有很好的内存容量且能耗更低，但其读写带宽、延迟以及写耐久相对DRAM较差。因此，对于软件，如何高效使用这些异构内存架构中的内存单元是一个很困难的问题。

**项目目标**：本项目需要设计和实现一种检测内存访问模式的二进制代码分析工具，通过识别不同的内存访问模式，能够更好地对软件中的数据对象在异构内存中的数据放置进行优化。同时提供用户友好的代码分析工具，指导开发人对代码进行数据放置优化。特别是对NVM+DRAM的异构内存系统，识别数据对象访问中的顺序读模式，并将该类型数据对象放置在NVM中，既可以享受NVM大容量、低功耗的优势，又不会对软件性能产生明显影响。其次，对于其它访问模式的数据对象，需要定量化其对NVM的性能敏感度，并依据性能敏感度制定数据对象在NVM和DRAM间迁移的优先级，指导运行时数据迁移，降低其对软件性能的影响。

**项目研发内容：**

1. 记录每个内存位置的load/store数量，并跟踪软件静态、动态创建的数据对象，从而得到每个数据对象的load数量热力图、store数量热力图、load/store比热力图等分析结果，从而获得数据对象的内存访问模式；
2. 对于CPU上的混合内存架构（如NVM、DRAM、HBM等），拟采用PEBS tracing采样得到访存地址，并同时通过ELF符号表分析、malloc函数类的封装等技术跟踪静态、动态构建的数据对象；
3. 对于GPU，拟采用基于CUPTI的动态插桩方法对并发执行的thread block进行采样得到访存地址，并对malloc类函数使用event进行跟踪得到动态构建的数据对象；

**项目技术路线：**

1. CPU：地址可以通过Intel PIN得到，潜在问题：可能需要单独实现定位到特定的data object的功能 （可以查一查有没有相关的库实现）
   1. 特定到data object：
      1. Static object: 可以参考CCTLib的cctlib.cpp中的compute\_static\_var函数实现，原理为读取elf中的符号表，从而定位静态变量和具体内存空间之间的映射关系；
      2. Dynamic object: 原理为实现一个malloc函数类的封装函数，来抓取数据对象大小以及开始地址，还可以通过backtrace得到该动态数据对象创建时的调用上下文信息；
      3. 其他包括栈上的变量都可以忽略

参考资料包括[1][2][3]

1. GPU: GPU上的混合内存架构：Shared memory, Global memory, Texture memory, constant memory, ...
   1. 通过CUPTI插桩获取地址等信息

参考资料包括[4]

**参考资料**

NVM+DRAM基础知识，参考[6]；

NVM上数据放置优化，参考[5]；

二进制代码分析和数据放置工具，参考[7][8]

[1] Intel® 64 and IA-32 Architectures Software Developer Manuals

[2] <https://github.com/CCTLib/cctlib>

[3] Chen Y, Peng I B, Peng Z, et al. ATMem: adaptive data placement in graph applications on heterogeneous memories[C]//Proceedings of the 18th ACM/IEEE International Symposium on Code Generation and Optimization. 2020: 293-304.

[4] <https://docs.nvidia.com/cupti/Cupti/index.html>

[5] Liu, Jiawen, et al. "Sparta: High-Performance, Element-Wise Sparse Tensor Contraction on Heterogeneous Memory."

[6] Patil, Onkar, et al. "Performance characterization of a DRAM-NVM hybrid memory architecture for HPC applications using Intel Optane DC Persistent Memory Modules." Proceedings of the International Symposium on Memory Systems. 2019.

[7] Yu, Chao, et al. "Lwptool: A lightweight profiler to guide data layout optimization." IEEE Transactions on Parallel and Distributed Systems 29.11 (2018): 2489-2502.

[8] You, Xin, et al. "ZeroSpy: exploring software inefficiency with redundant zeros." 2020 SC20: International Conference for High Performance Computing, Networking, Storage and Analysis (SC). IEEE Computer Society, 2020.

[9] Intel PIN. https://software.intel.com/content/www/us/en/develop/articles/pin-a-dynamic-binary-instrumentation-tool.html