Heap Architectures for Concurrent Languages using Message Passing

Authors describe an analysis-driven storage allocation scheme for concurrent languages that use message passing with copying semantics. The basic principle is that in such a language, data which is not part of any message does not need to be allocated in a shared data area. This allows for deallocation of threadspecific data without requiring global synchronization and often without even triggering garbage collection. On the other hand, data that is part of a message should preferably be allocated on a shared area, which allows for fast (O(1)) interprocess communication that does not require actual copying. In the context of a dynamically typed, higher-order, concurrent functional language, we present a static message analysis which guides the allocation. This paper shows performance evaluation, conducted using an industrial-strength language implementation, the analysis is effective enough to discover most data which is to be used as a message, and to allow the allocation scheme to combine the best performance characteristics of both a process-centric and a shared-heap memory architecture.

Reference:

http://www.fantasi.se/publications/ISMM02.pdf