Parallel Genetic Algorithm Taxonomy Cheat sheet

Master-Slave Parallelization (Distributed Fitness Evaluation) The master node stores the population, than fractions of the population are assigned to the individual nodes and evaluated in parallel. Communication only occurs when the population is sent to the nodes then results are sent back. The two types are synchronous where the master waits for the evaluations to complete and asynchronous where it continues after a fraction of the results return. **Pros:** Easy to implement. Good speed up if communication cost does not exceed computation speedup. **Cons:** Synchronous needs to wait for the entire population to be analyzed. Asynchronous the algorithm becomes more complex and it is difficult to measure its effects.	**Static subpopulations with migration** Uses multiple demes (Subpopulations) then has individuals migrated between the demes. The individual demes are separated from each other (geographic isolation) and competition only occurs within that deme. This is the most popular and heavily studied algorithm. **Pros:** Easy to implement, just take a simple GA and put it on multiple nodes. Then implement a migration algorithm. **Cons:** Difficult to scale as the number of subpopulations and individuals increase.
Dynamic demes (Dynamic Overlapping Subpopulations) This is a method that reduces the waiting time seen in synchronous master-slave parallelization. Individual demes are created and can be scaled with more demes or more individuals. These demes are then split into demes that can be processed right away reducing the wait times. This algorithm can be run on shared or distributed parallel machines. **Pros:** Easy to implement, scalable **Cons:** New area of research	**Massively parallel genetic algorithms** Like a fine grained and static overlapping algorithm. Use a lot of individuals and a lot of demes, but use a data structure such that the demes overlap with their neighbors (A grid is common). It's tricky to implement due to the complexity with the synchronization and the huge amount of communication necessary. **Pros:** Well suited for massively parallel computers. **Cons:** Can be implemented on a small cluster but has an extremely high communication cost.
Static overlapping subpopulations (without migration) The entire population consists of individual demes but some individuals are placed in multiple demes (overlapping). **Pros:** In a shared memory model it can be implemented easily. **Cons:** Synchronization becomes an issue for the individuals in the overlapping areas.	**Parallel messy genetic algorithms** Initial populations are created in a primordial phase using partial enumeration. The individuals are then reduced in a tournament style selection. In the next phase partial solutions are then mixed together and evaluated.
Parallel steady-state genetic algorithms A single population is created then individuals are gradually introduced over time. The evaluation functions are done in parallel.	**Hybrid methods (e.g. static subpopulations with migration, with distributed fitness evaluation within each subpopulation)** Combine features of each into one algorithm.

Information obtained from: Parallel Genetic Algorithm Taxonomy by Mariusz Nowostawski and Riccardo Poli