(Here and in the following, by R_W we denote the set of all requested services in a set of areas $W \subseteq V$, i.e., $R_W = \bigcup_{w \in W} R_w$. Thus, in particular, R_V is the set of *all* requested services.)

This suggests two different types of ensemble packing heuristics which we would expect to work reasonably well with sparse and dense graphs, respectively. For sparse graphs, we will pack the individual requirement sets independently of each other. For dense graphs, we will pack the entire collection of requested services, and then assign to each vertex those ensembles which are needed to satisfy the supply requirements at that vertex. For an arbitrary graph, we might try both approaches and see which one works best.

Now we still face the problem that the bin packing problem is itself NP-complete. Fortunately, however, it is possible to obtain fairly good approximations of optimal bin packings using the following kind of "greedy" procedure known as the *First-Fit* (*FF*) bin packing algorithm. The algorithm is invoked with the set X to be packed, element sizes μ_x, $x \in X$, and bin size M (where we assume that $M \geq \mu_x \; \forall \; x \in X$, since otherwise no solution exists). It starts out with the empty packing $\mathcal{B} = \emptyset$ and considers the elements of X in some given order. For each $x \in X$, it then adds x to the "first" bin $B \in \mathcal{B}$ into which it "fits," i.e., $\mu_B + \mu_x \leq M$; if no such bin exists, it adds a new bin B to $\mathcal{B}$ and makes x its single member. (The bins are always considered in the order in which they were added to the packing.)

It can be shown that, regardless of the chosen element order, the size of a packing computed with the first-fit packing algorithm always satisfies $|\mathrm{FF}(X, M)| \leq \frac{17}{10} p_M(X) + 2$, and hence the asymptotic deviation from the optimum is $\leq 70\%$. If we order elements by decreasing sizes, then the algorithm even achieves an asymptotic performance ratio of 11/9, corresponding to an asymptotic deviation of at most about 22% [6]. This is fairly good and sufficient for most applications. (To further improve on this – at the cost of a considerable increase in running time – there also is an asymptotic "fully polynomial time approximation scheme" based on linear programming methods [9, p. 1574].)

We can apply the first-fit algorithm to ensemble packing using the two strategies sketched out above, as follows:

Simultaneous First-Fit (SFF) Algorithm:

$$\mathcal{B}_v = \mathrm{FF}(R_v, M) \; \forall \; v \in V, \tag{5}$$

Global First-Fit (GFF) Algorithm:

$$\mathcal{B}_v = \{B \in \mathrm{FF}(R_V, M) : B \cap R_v \neq \emptyset\} \; \forall \; v \in V. \tag{6}$$

Incidentally, the two algorithms also correspond to the two solution approaches taken in our example in Section 3. The SFF algorithm computes *strict* solutions (ensemble assignments exactly satisfying the requirements), aiming at "local optimality" by avoiding over-supply. In contrast, the GFF algorithm strifes for "global optimality," producing *overlap-free* solutions in which the constructed ensembles are mutually disjoint (i.e., each