G be a one-vertex graph.) So we know that Problem 1 is not only NP-complete, but also difficult to approximate, and hence we will be interested in heuristic solutions.

How can we approach this problem? We have already mentioned that the problem reduces to an ordinary graph coloring problem once we have obtained a suitable ensemble assignment. Thus a straightforward approach is to solve the problem in two optimization stages:

- Find an admissible ensemble assignment $\mathcal{B}$.
- Color $G^{\mathcal{B}}$ using some heuristic graph coloring procedure.

For the second stage, a plethora of different graph coloring algorithms is already available. We can simply plug any of these methods into our ensemble planning procedure. In fact, it turns out that fairly simple "sequential" coloring methods like Brélaz' DSATUR [1] or Matula/Beck's smallest-last algorithm [10] usually perform very well on the kind of "geometric" graphs arising as models of broadcast networks. So we can concentrate on the first step, which obviously is a kind of simultaneous bin packing problem. The difficult part here is to devise a packing strategy which reduces the global number of required *colors* instead of merely optimizing the packings for individual areas. Of course no single strategy will work with all area graphs equally well, so let us take a look at the two extreme cases:

- *Independent (edgeless) graphs.* In this case the problem decomposes into $|V|$ independent bin packing problems, one for each vertex of G, and we have that $\chi_M^R(G) = \max\{p_M(R_v) : v \in V\}$.
- *Complete graphs.* In this case, all distinct ensembles will have to be assigned different frequency blocks, hence $\chi_M^R(G) = p_M(R_V)$.

(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