When the number of arms in a contextual bandit problem is large, traditional approaches for selecting arms may become inefficient due to computational complexity and memory constraints. Here are some strategies to handle this situation:

1. **Sparse Representations**: Use sparse representations for the arms and their features. Instead of storing all feature values explicitly, represent them in a sparse format, where only non-zero elements are stored along with their indices. This reduces memory requirements and computational complexity.
2. **Feature Selection or Dimensionality Reduction**: Apply techniques such as feature selection or dimensionality reduction to reduce the dimensionality of the feature space. This can help in reducing computational complexity while retaining relevant information for decision-making.
3. **Randomized Arm Selection**: When the number of arms is very large, a simple strategy might be to randomly select arms for exploration. Although this may not be optimal in terms of learning efficiency, it can be computationally feasible and can provide reasonable performance, especially in scenarios where exploration is less critical.
4. **Clustering or Hashing**: Group similar arms together using clustering techniques or hashing functions. Instead of considering each arm individually, explore arms within clusters or hash buckets. This reduces the effective number of arms considered during exploration, making it more computationally tractable.
5. **Approximate Algorithms**: Use approximate algorithms for arm selection that provide near-optimal solutions with reduced computational complexity. These algorithms often sacrifice some level of optimality for efficiency, making them suitable for large-scale problems.
6. **Online Learning Algorithms**: Explore online learning algorithms designed specifically for large-scale problems. These algorithms are optimized for efficiency and scalability and often employ techniques such as stochastic gradient descent, online convex optimization, or bandit-specific algorithms tailored for large-scale settings.
7. **Parallelization**: Exploit parallel computing techniques to distribute the computation across multiple processors or machines. This can help in scaling up the computation to handle large numbers of arms efficiently.
8. **Incremental Learning**: Instead of processing all arms simultaneously, consider incremental learning approaches where arms are processed in batches or streamed one at a time. This allows for continuous learning and adaptation while mitigating the computational burden.

By applying one or more of these strategies, it's possible to handle large numbers of arms efficiently in contextual bandit problems while still achieving reasonable performance in terms of exploration and exploitation. The choice of approach depends on the specific characteristics of the problem and available computational resources.