To make a functional program scalable

1. Use a modular design: Break your program down into smaller, reusable modules that can be easily distributed across multiple servers or nodes. This can help improve performance and scalability.

2. Use message queues: Use message queues to decouple modules and enable asynchronous communication between them. This can help improve throughput and scalability.

3. Use distributed computing frameworks: Use distributed computing frameworks such as Apache Spark, Apache Flink, and Hadoop to distribute computation across multiple nodes. This can help improve performance and scalability.

4. Use caching: Use caching to reduce the amount of computation and I/O required for frequently accessed data. This can help improve performance and scalability.

5. Use load balancing: Use load balancing to distribute traffic across multiple servers or nodes. This can help improve performance and scalability.

6. Use horizontal scaling: Use horizontal scaling to add more servers or nodes to handle increasing load. This can help improve scalability.

7. Optimize algorithms and data structures: Optimize algorithms and data structures to reduce the amount of computation and memory required. This can help improve performance and scalability.

8. Use cloud services: Use cloud services such as AWS, Google Cloud, and Azure to automate scaling and reduce operational overhead. This can help improve scalability and reduce costs.

By following these guidelines, you can design and deploy a functional program that is scalable and can handle increasing load as your application grows. It's important to keep in mind that scalability is not a one-time effort, but rather a continuous process of optimization and improvement as your application evolves over time.