As it becomes more burdensome to run nodes fewer people will decide to. This has a couple negative effects:

- Security is reduced as more people trust providers such as Infura
- Developing novel eth applications is made harder, as developers are stuck with the interface Infura provides.
- The incentives to write and run a non-conforming client (such as one which threw away historical chain data after it had verified the correct chain) become greater, which would threaten the health of the network.

Ethereum relies on the charity of people who run nodes; by allowing the disk requirements to increase we're rewarding that charity by asking for ever larger donations!

I've listed some options below but I'm sure I've missed something, are there more/better ideas?

1. Stateless clients. This is the most thorough solution. It users were responsible for maintaining their own state then using ethereum would become harder (you definitely don't want to lose that state!), but running a node would become much easier. Nodes only need to sync the chain data.
2. SeF proposes using authenticated fountain codes to reduce node storage requirements. Essentially, after verifying the full chain each node would drop 9/10ths of it (or whatever ratio you like). They do this in such a way that a newly bootstrapping node only needs to talk to ~10 nodes in order to download the full chain. Even halving the storage requirement would be useful! And doing so would mean a node joining the network would need to sync against existing nodes.
3. Chain pruning is much simpler than SeF but I think a little more dangerous, it opens up the possibility that new full-syncing nodes can't find some parts of the chain.
4. Some kind of incentivization scheme.
5. Storing the state in swarm. It worries me because any system which concentrates responsibility for storing each part of the chain into a small part of a DHT means an attacker only needs to DOS one part of the DHT to prevent new nodes from joining the network.
6. State Rent. This work seems to have been deferred in favor of stateless clients?
7. I think there was a proposal for a zero knowledge proof? A proof that a given block has some total difficulty would allow nodes to join without downloading any chain data with similar security properties to how fast sync works now. I'm skeptical this is possible but it would be great!