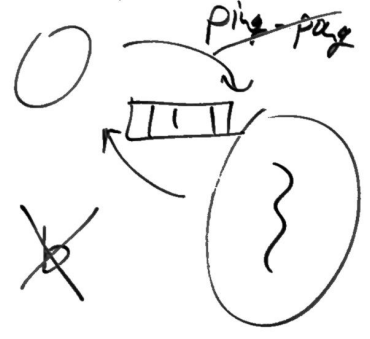
```
val b : Box[C]
val l : List[Int]
finish {
  future {
    b.open ...
    l.map(x => x+1)
  }
  future {
    :
  }
}
```

Spores



Sarina

```
val b
val l : b : Q▷ Box[C]
val task1 = new Task(b)
val task2 = new Task2(b)
    new Task4(b1, b2)

finish {
  future(task1) { task1 : Task^{Q1, Q2}
    // Perm[Q] ∉ Γ'
    future(task2)
      await(task1) {...
    future(task3) {
    }
  }
}
```

```
task Task4(b : τ) = (
box[C] { b =>
```

Q Box

map-reduce

b : Q▷ Box[C]
∈ Γ
Perm[Q] ∈ Γ

```
P.send(b) {
  // Γ' ∌ Perm[Q]
}
}
```