```
val t = task(b, x => ...)

finish {
    let x = async(t) {s} in
```

come
task perm.

```
[capture(x.f, b) {...}
}
```

modify
box of task

```
finish {
    let _ = async(t) {s'}
```

```
finish {
  let x = capture...
  in y }
```

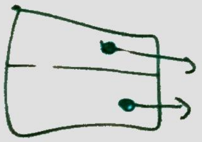
cont

```
def f(b): let = {
```

finish { ... finish{ ..capture } ...cap...

```
let x = capture(x.f, b) {cont}
∅
```

```
finish {
    async(t) {
        let ..
        async(t_2) {
            s
        }
    }
}
```



```
def cap(x: Box[T]
        y: Box[T]) -
```