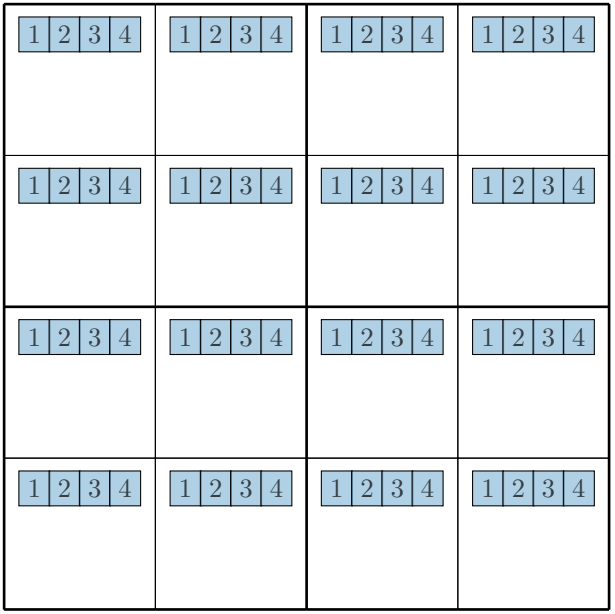
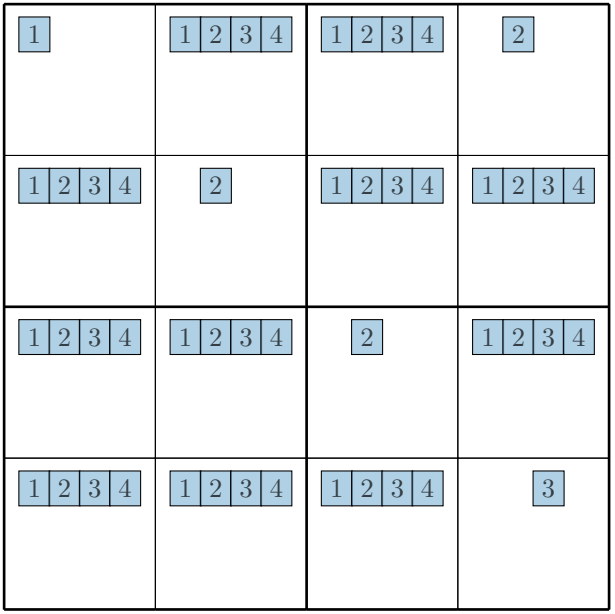


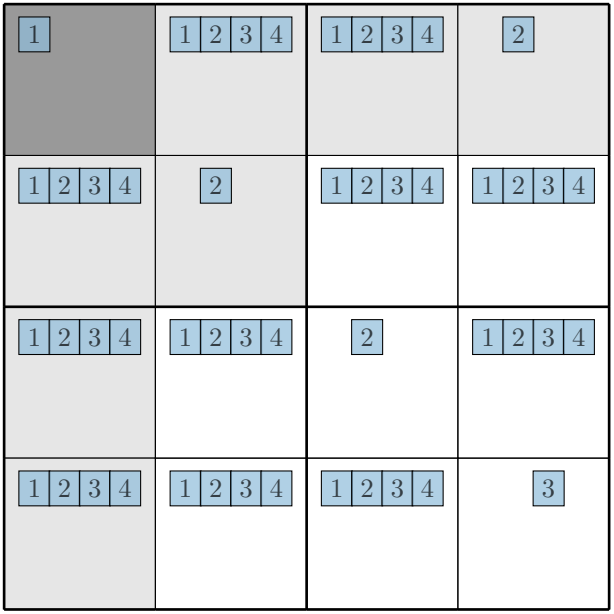
`solve(problem`: unary constraints)

- `domains = init_domains()`
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains = copy(domains)`
    - `spot, num = search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains = backtrack(stack)`



`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

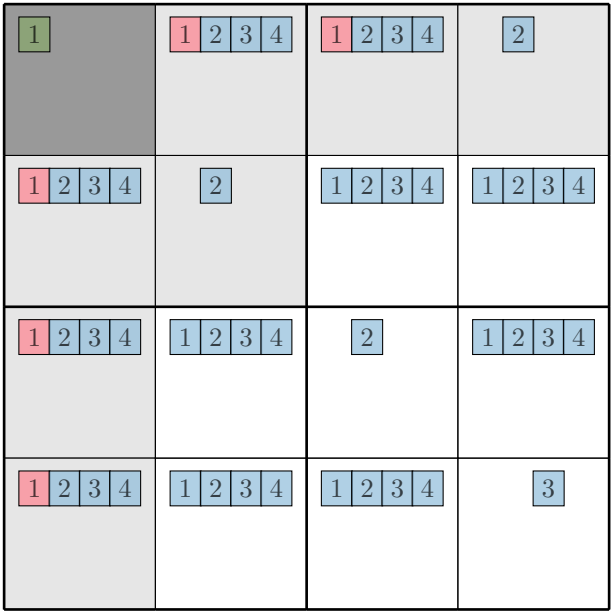


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

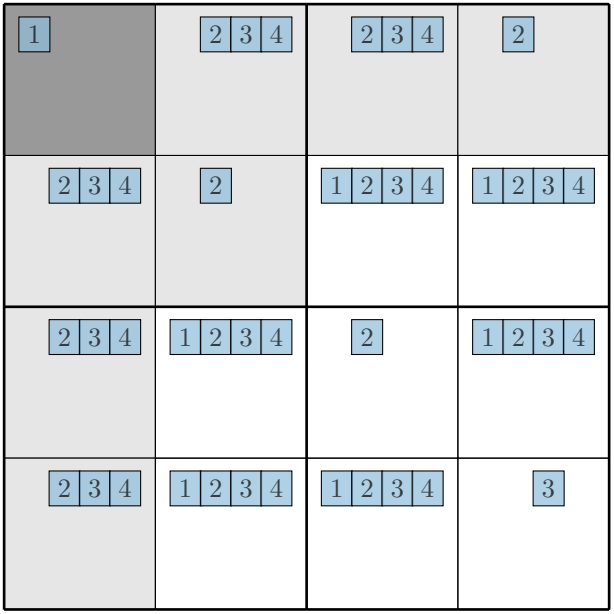


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

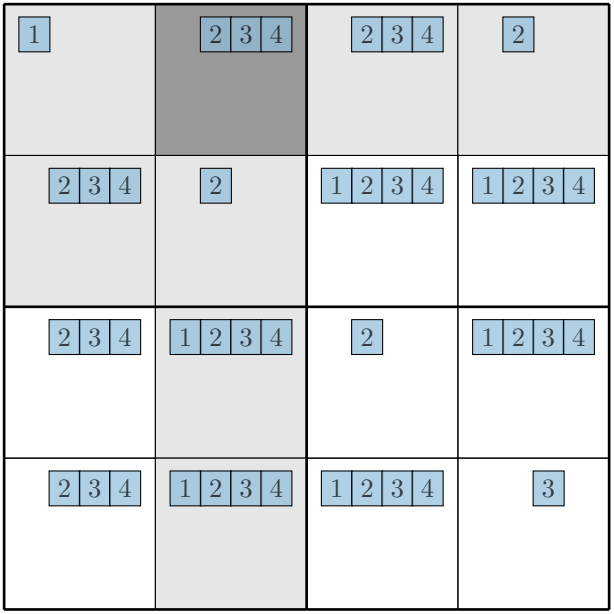


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

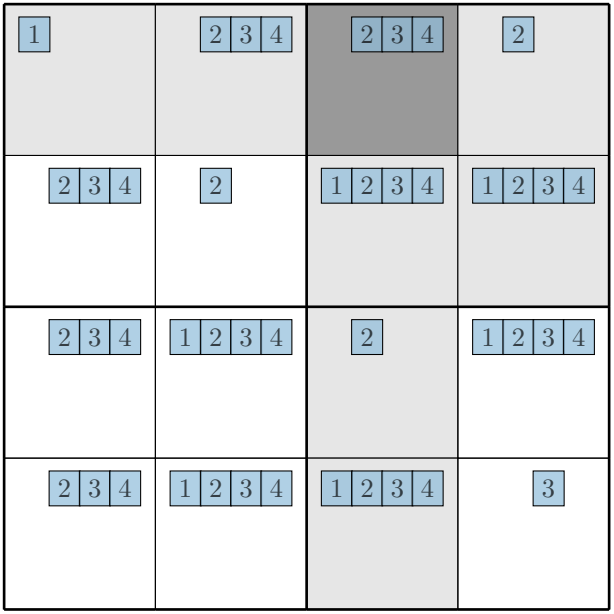


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

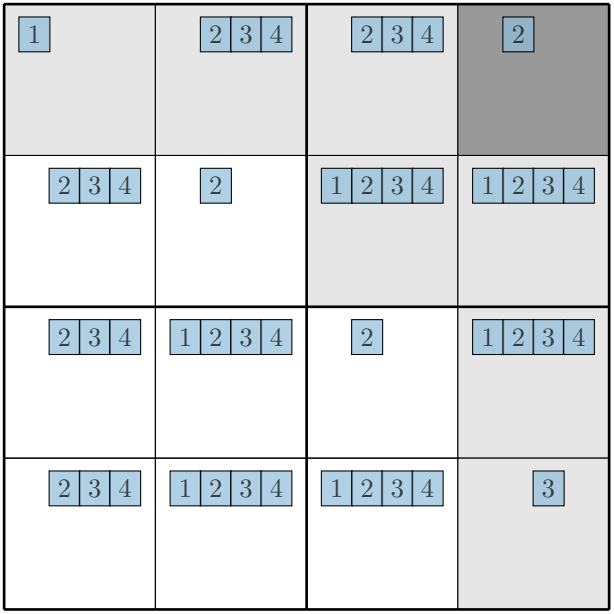


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

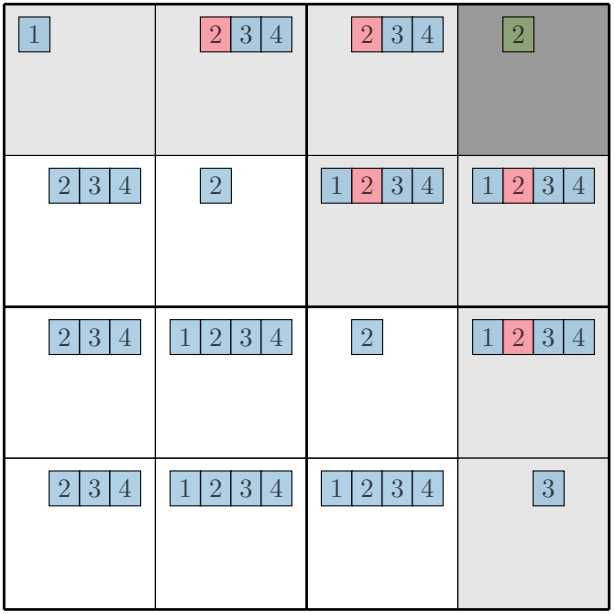


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

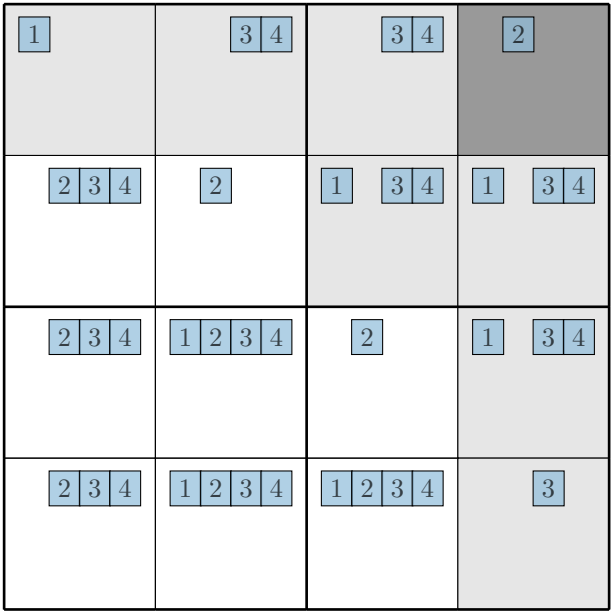


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

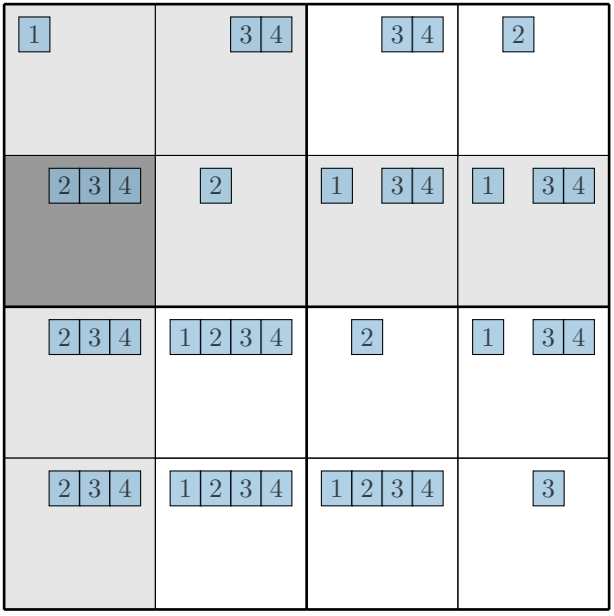


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

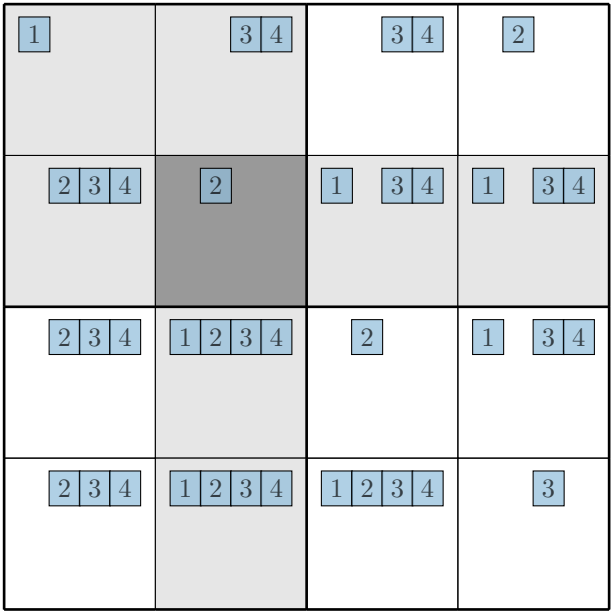


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

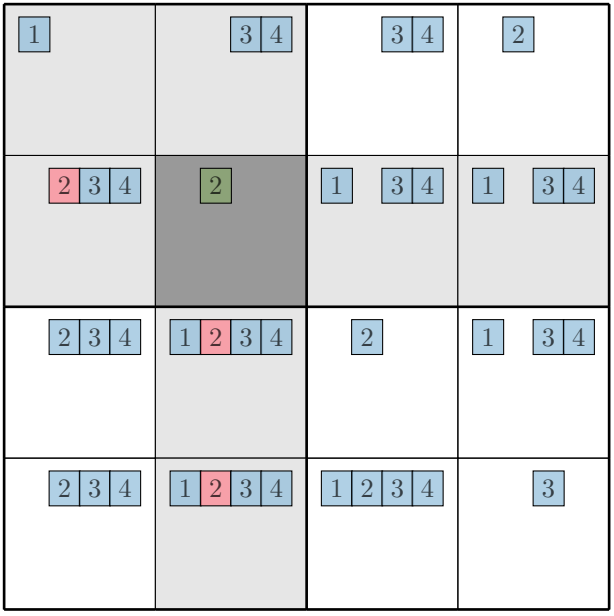


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

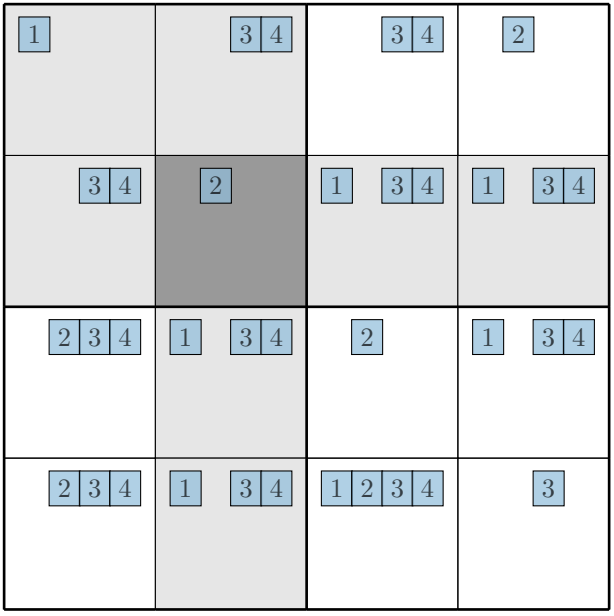


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

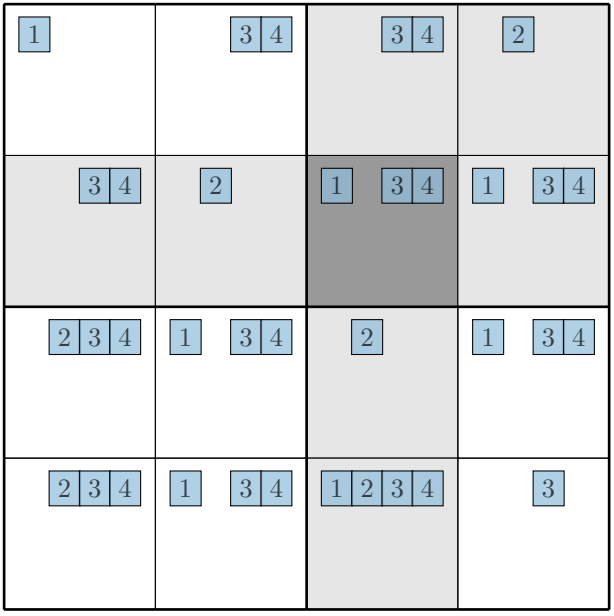


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

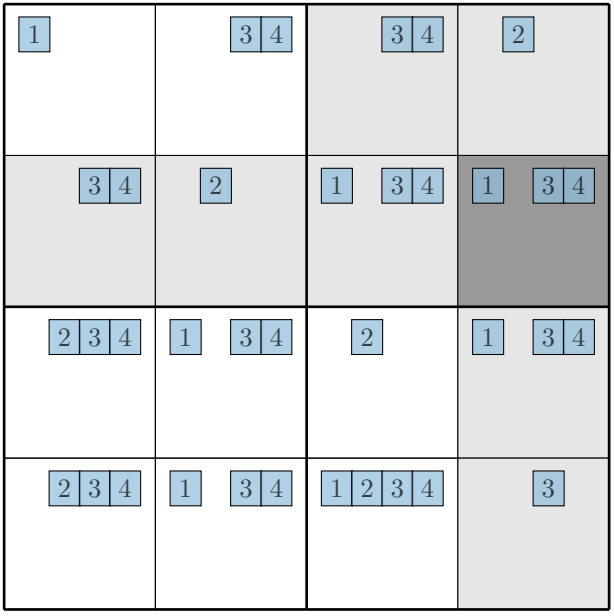


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

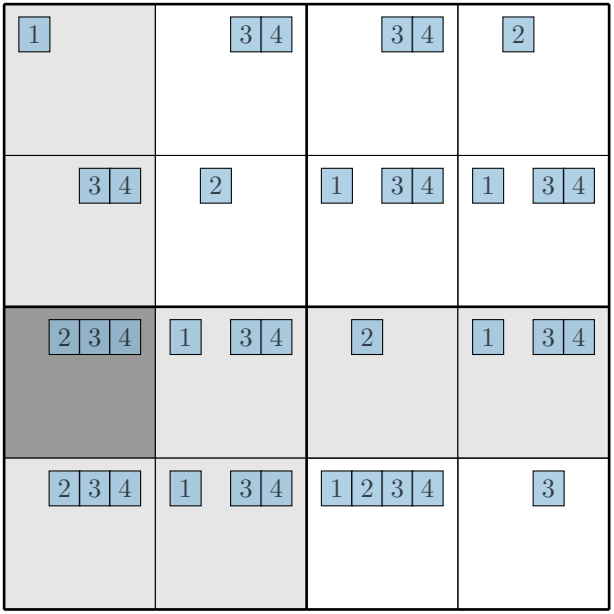


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

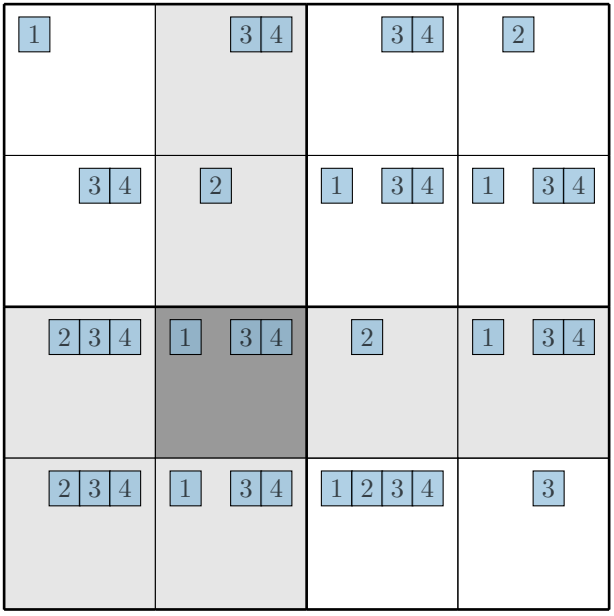


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

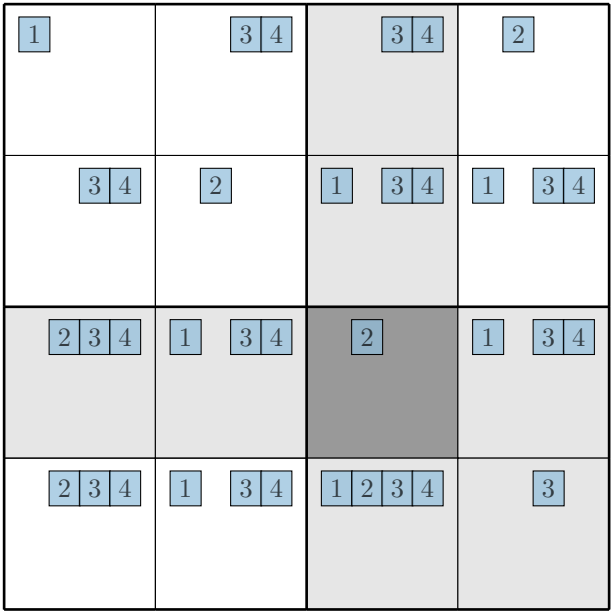


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

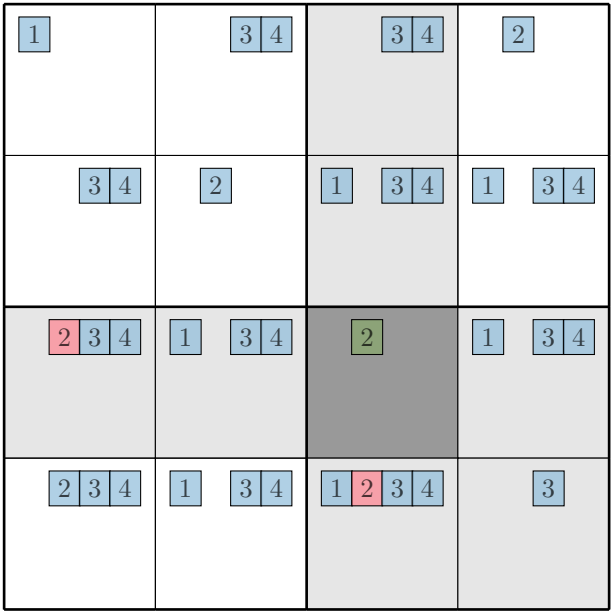


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

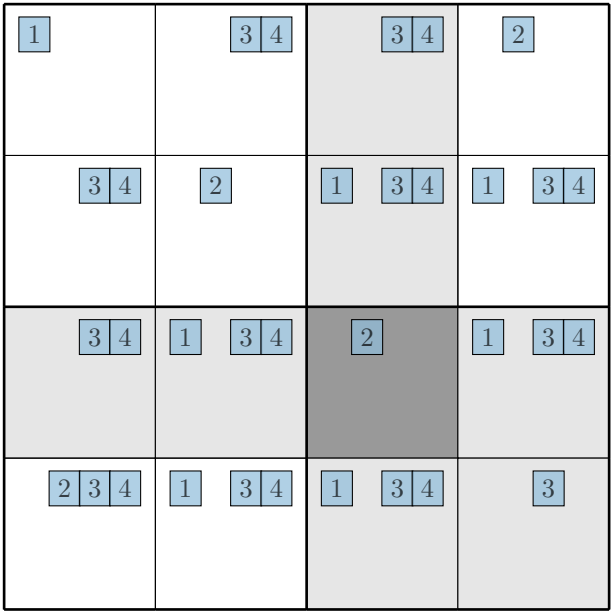


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

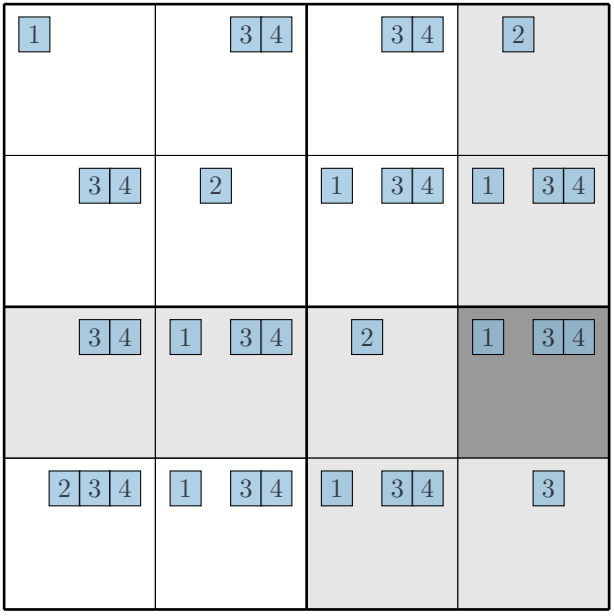


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

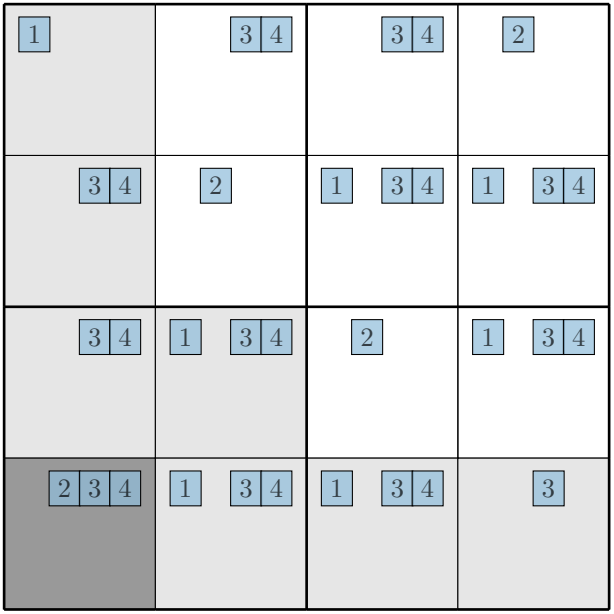


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

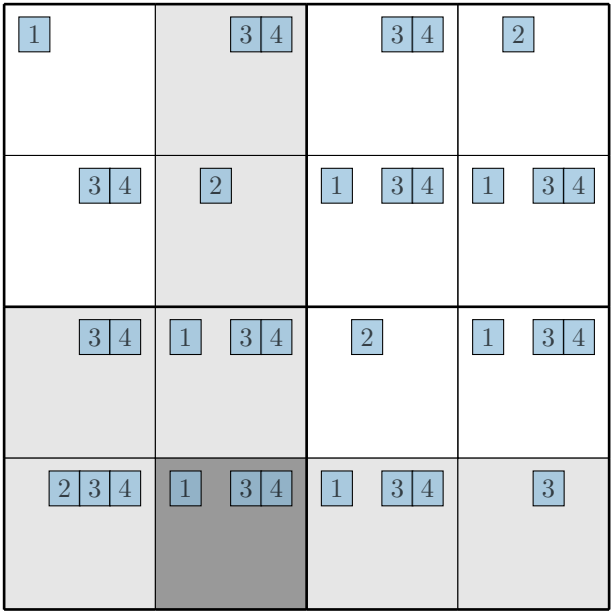


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

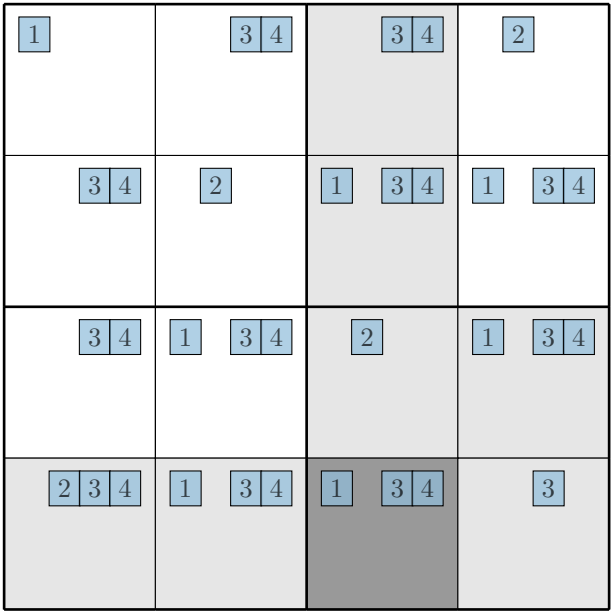


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

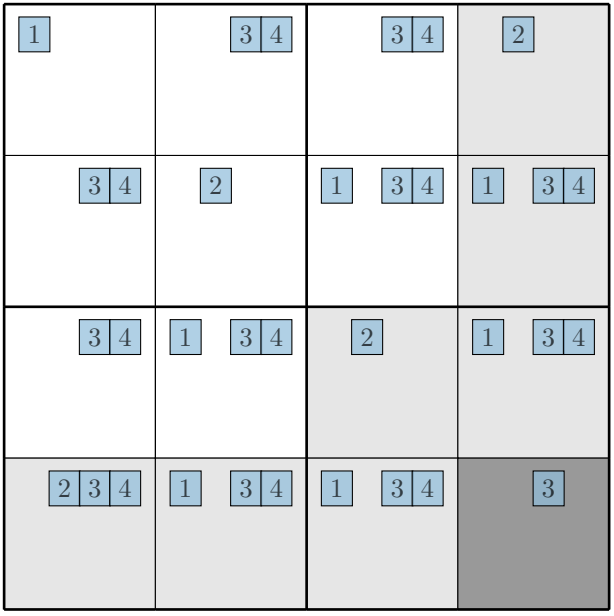


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

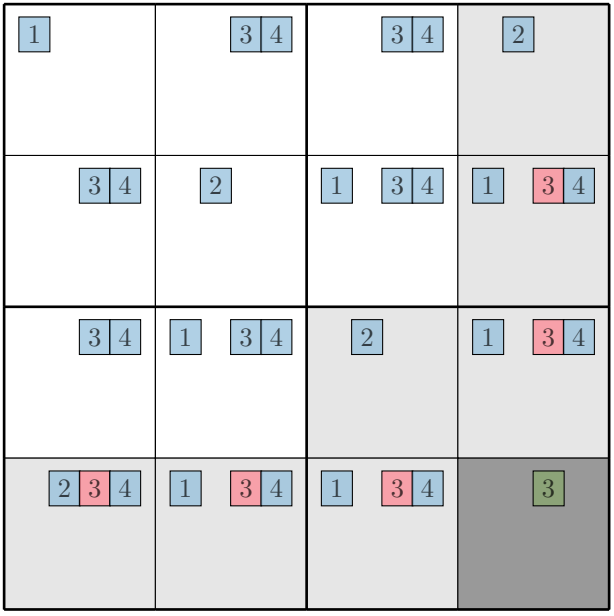


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

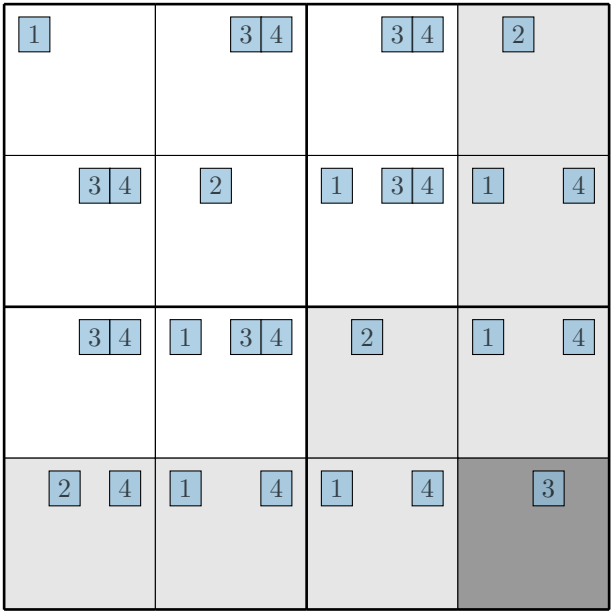


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

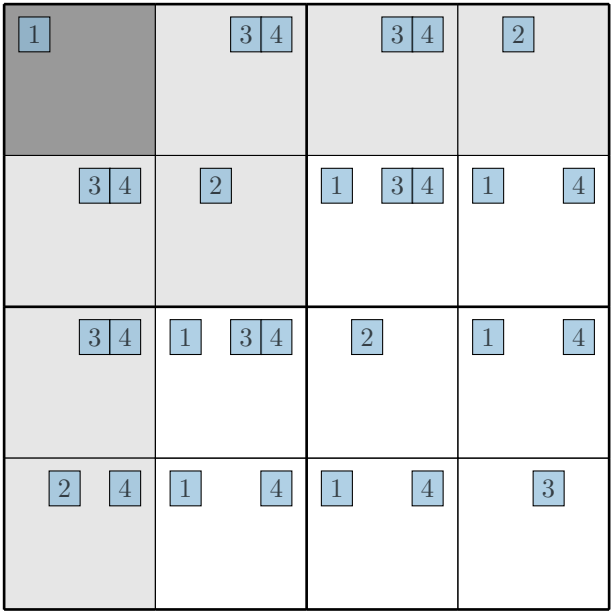


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

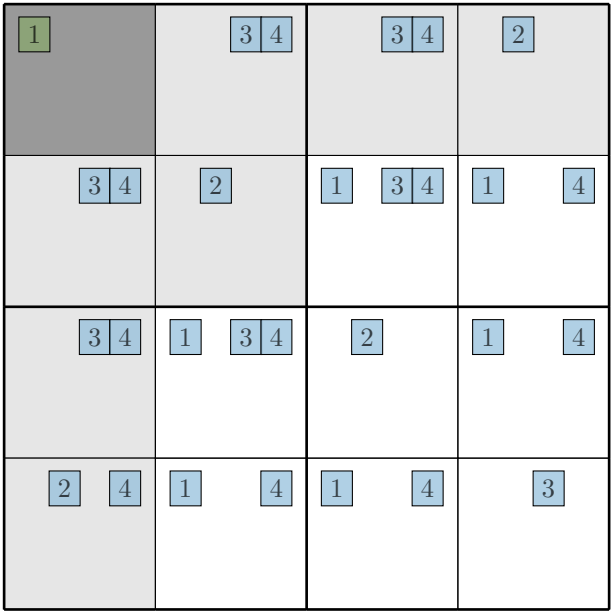


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

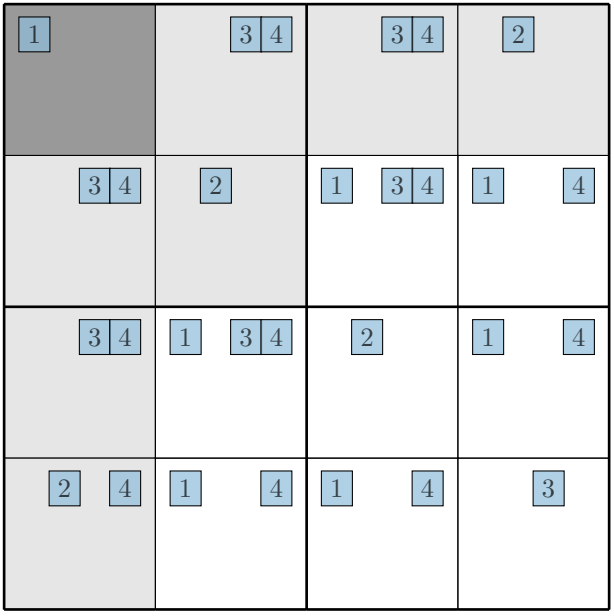


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

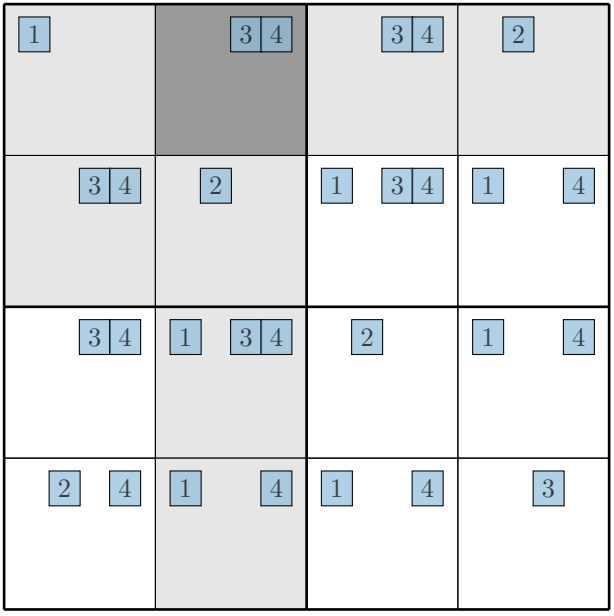


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

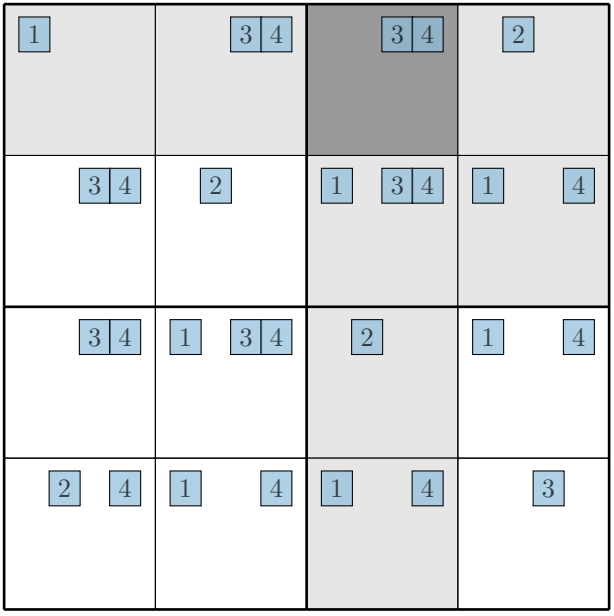


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

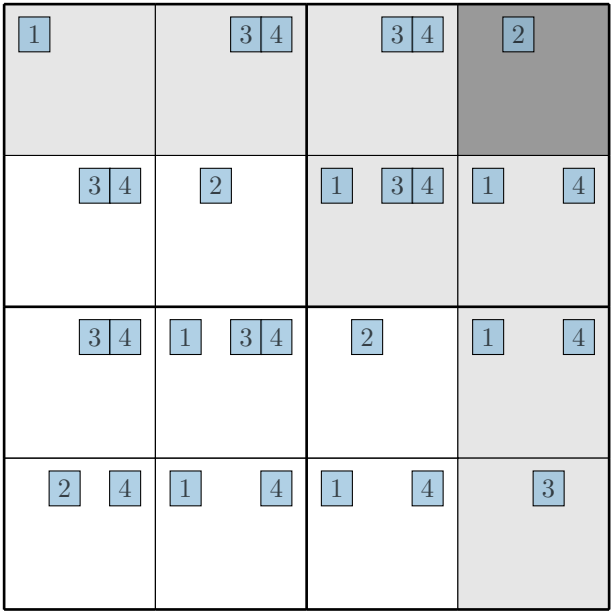


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

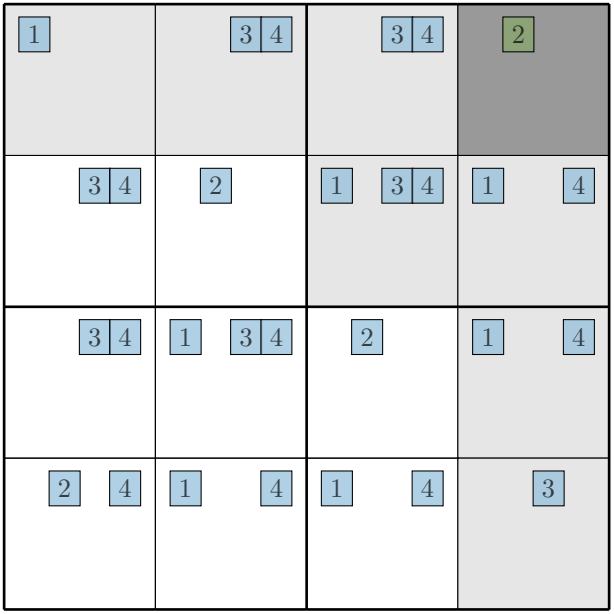


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

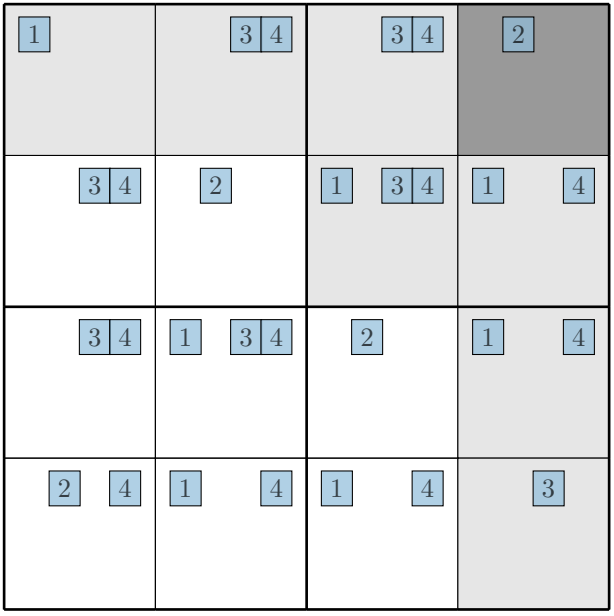


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

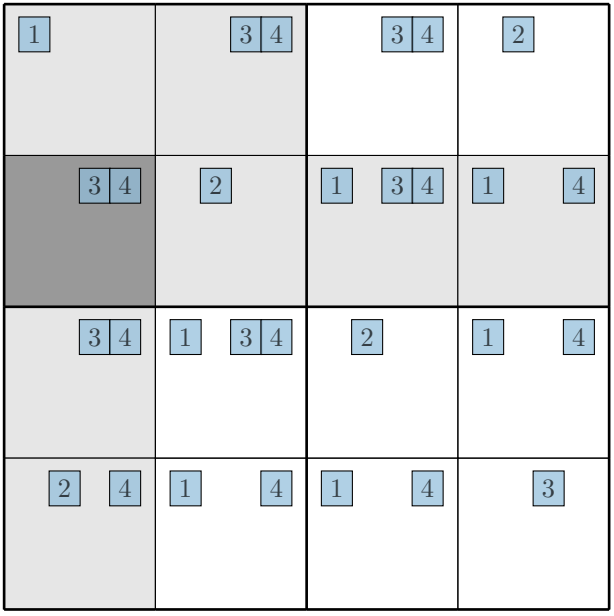


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

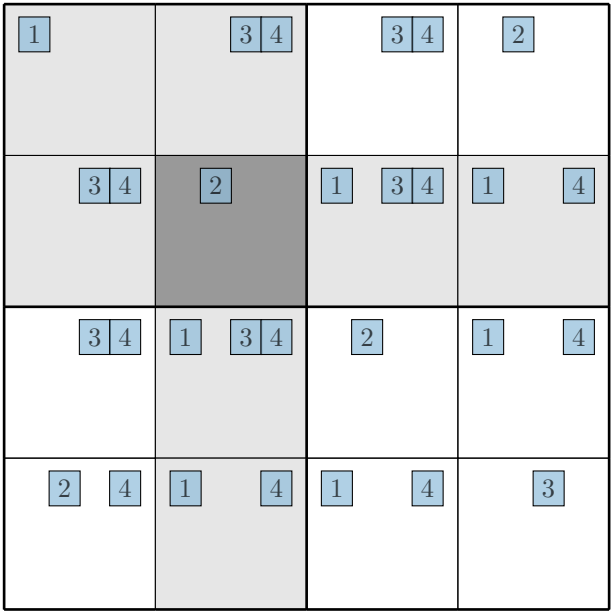


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

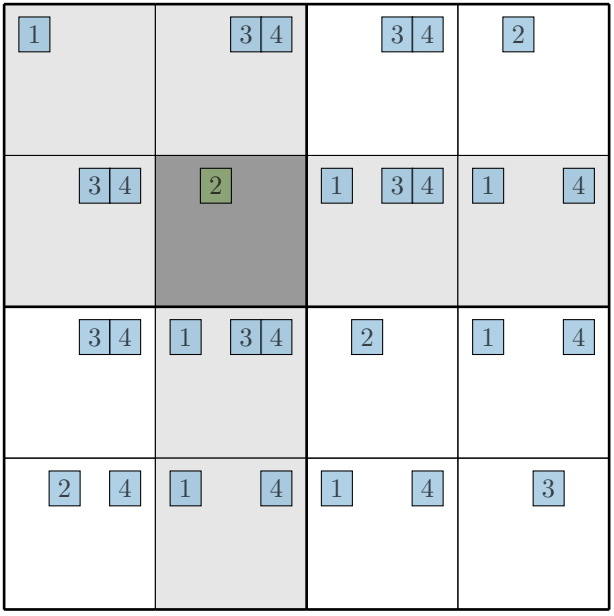


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

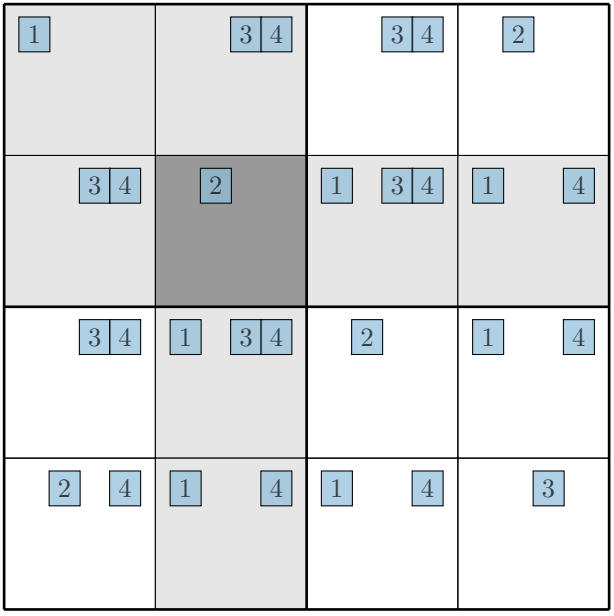


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

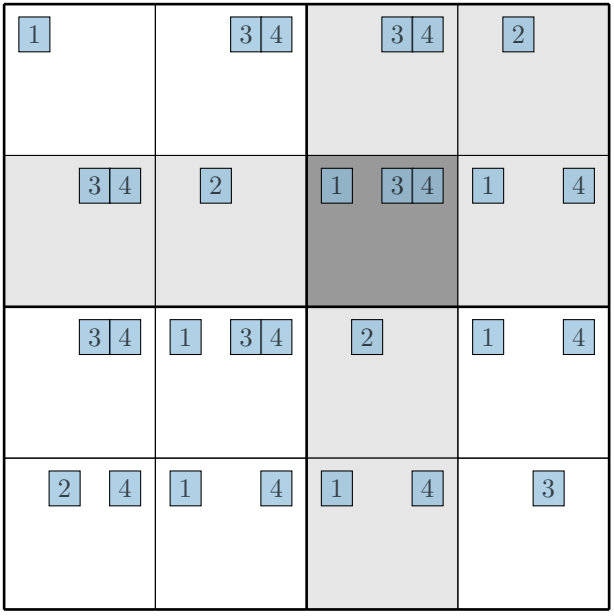


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

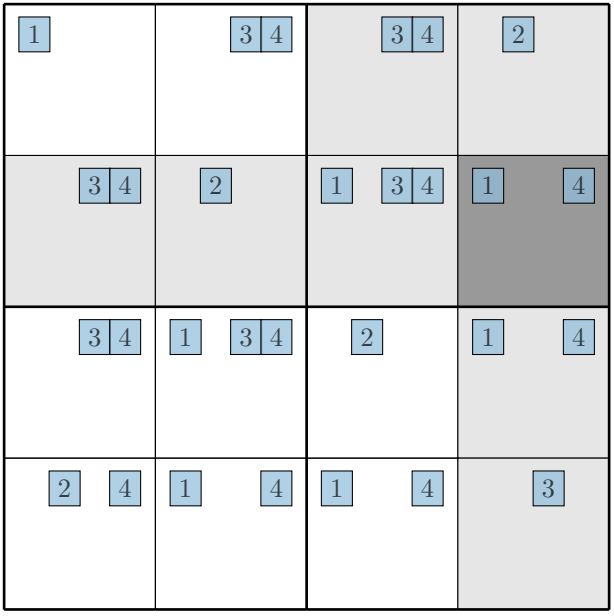


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

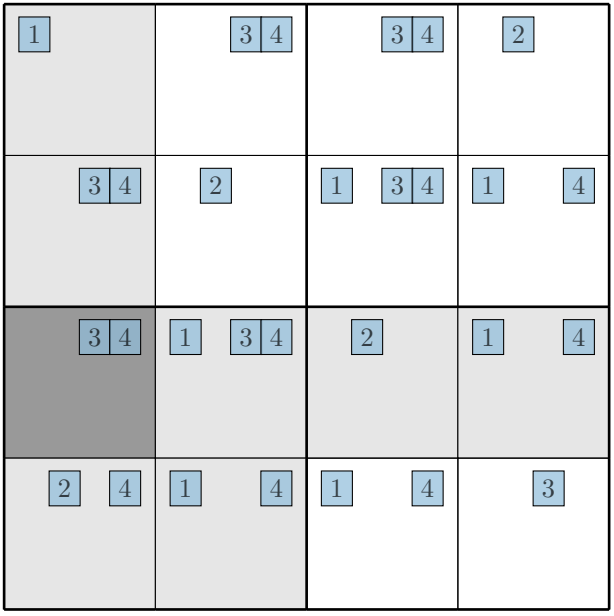


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

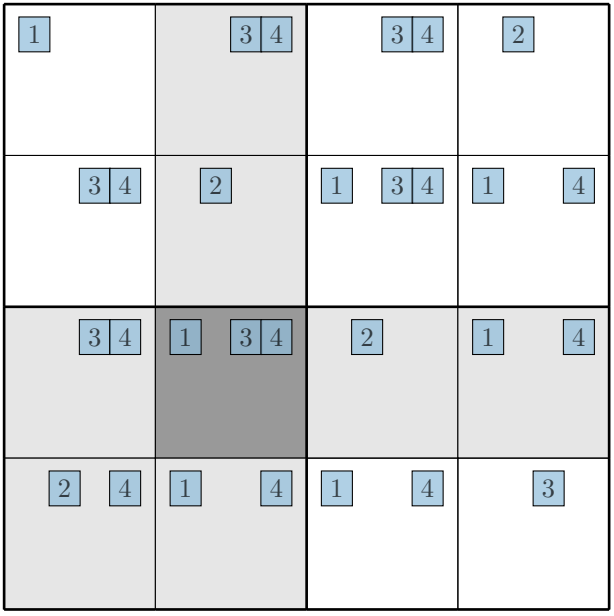


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

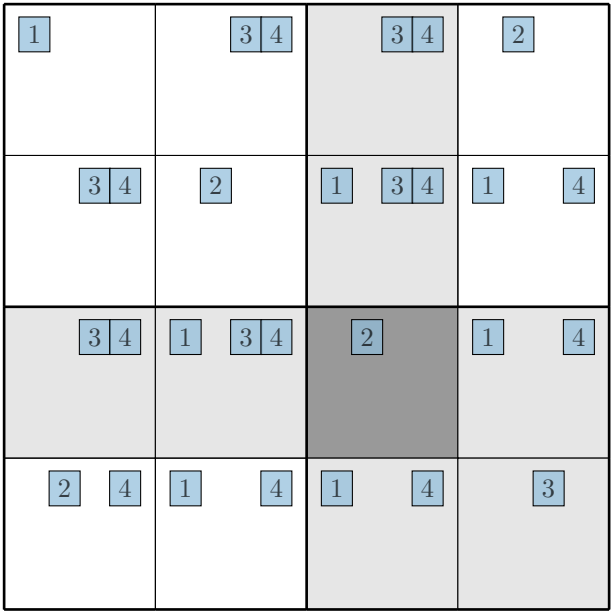


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

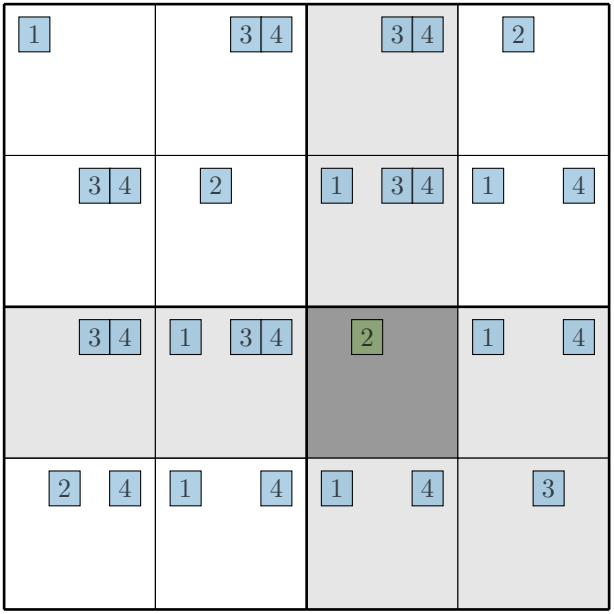


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

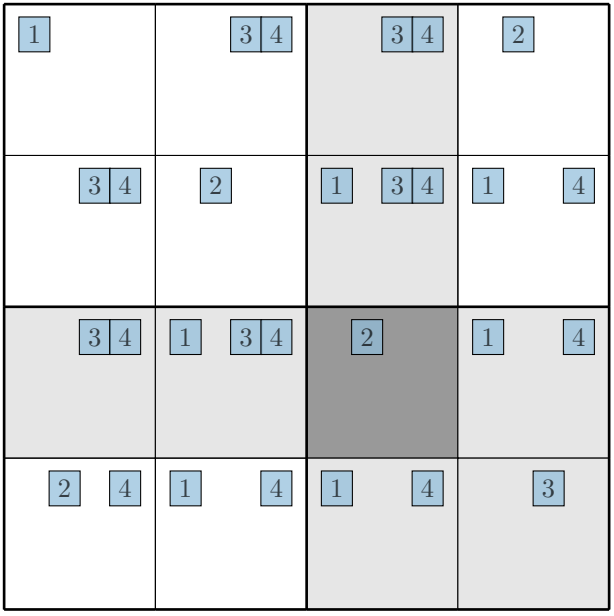


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

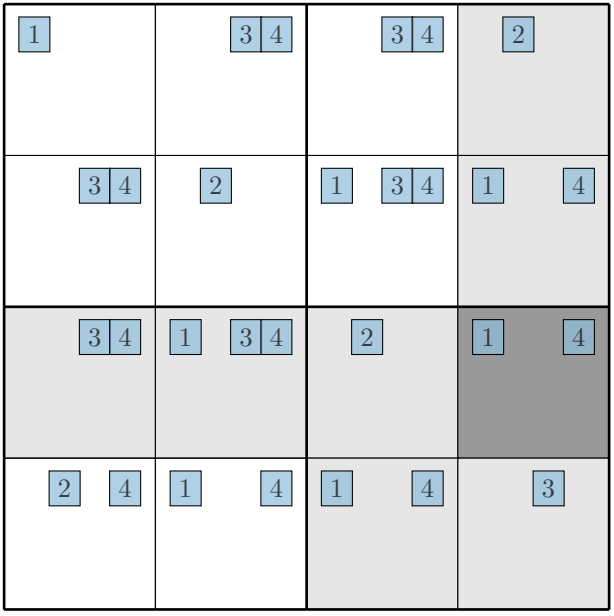


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

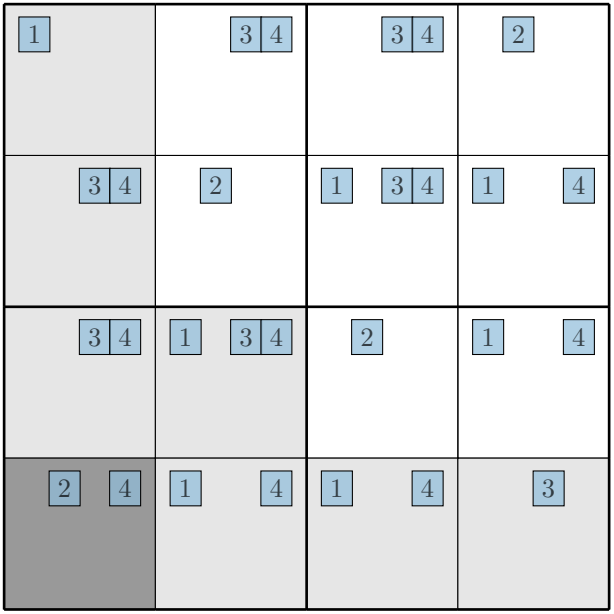


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

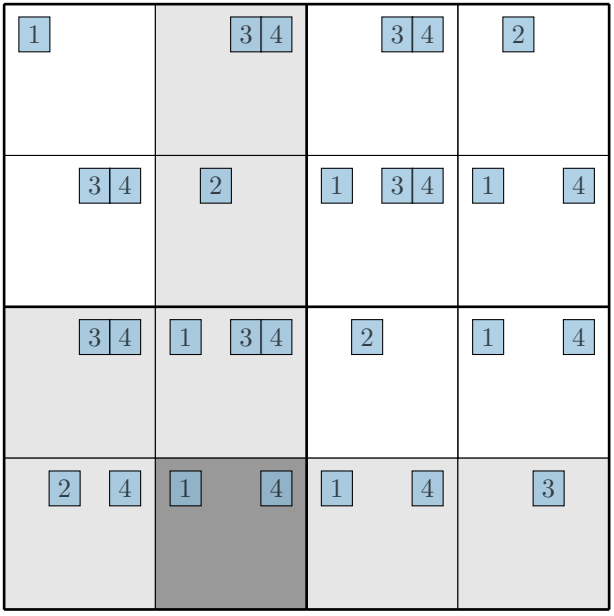


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

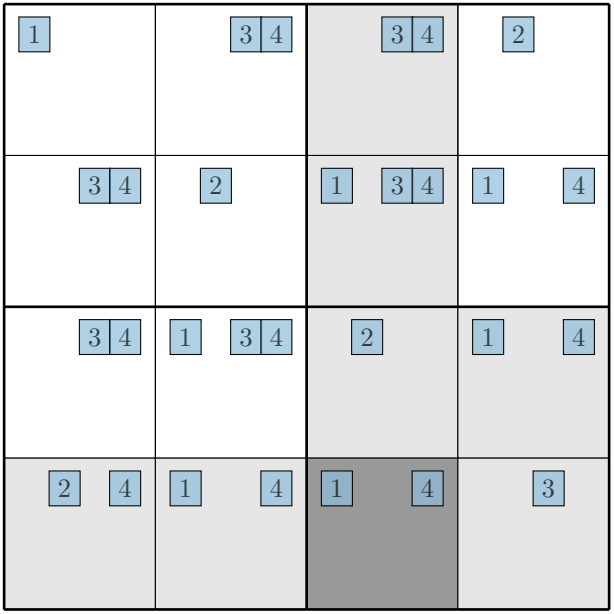


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

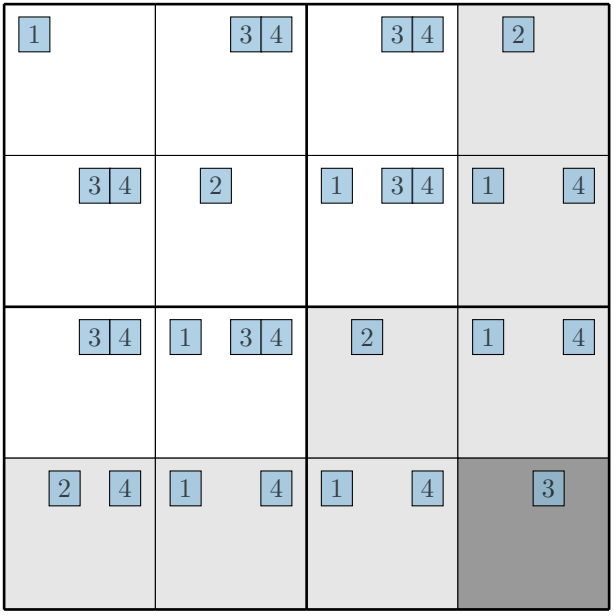


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

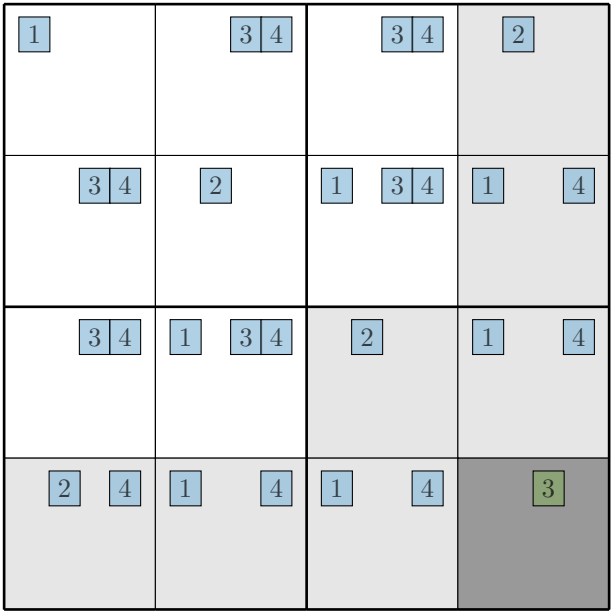


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

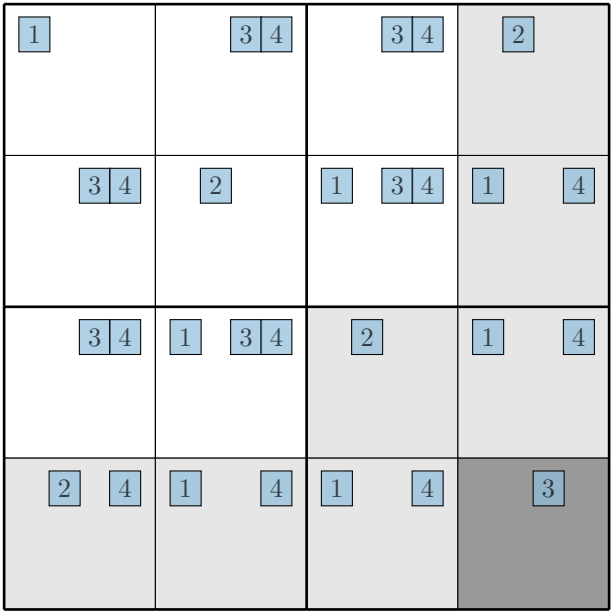


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

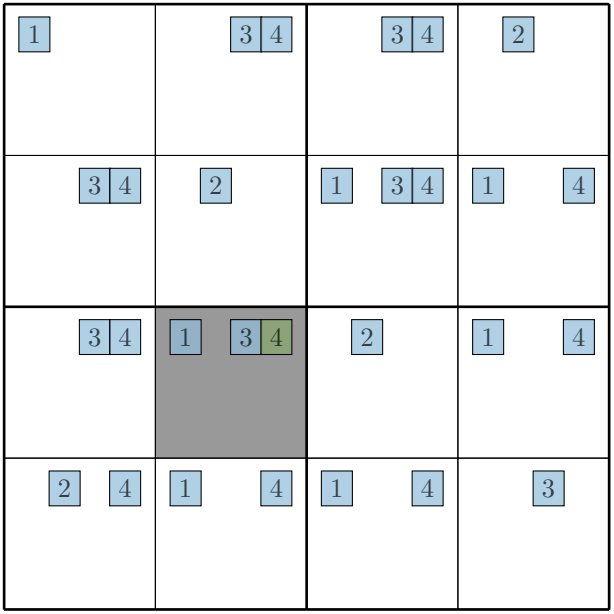


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

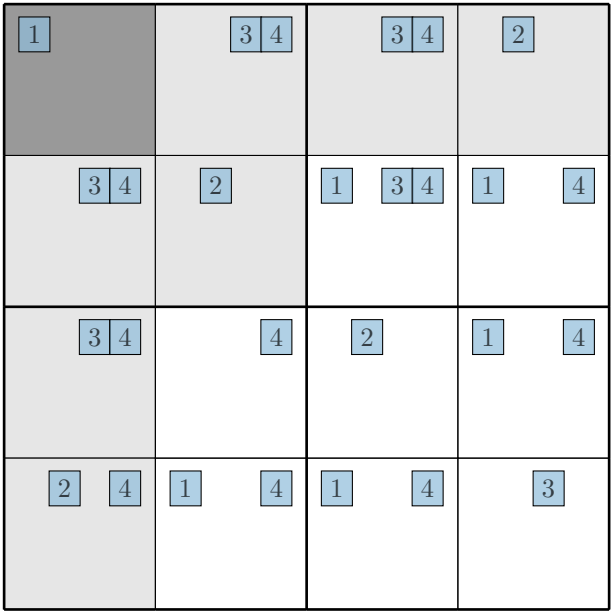


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`search`(`domains`: tile domains)

- `spot` = an unassigned spot (according to `domains`)
- `num` = a possible num for `spot` (according to `domains`)
- restrict the domain of `spot` to `num`.

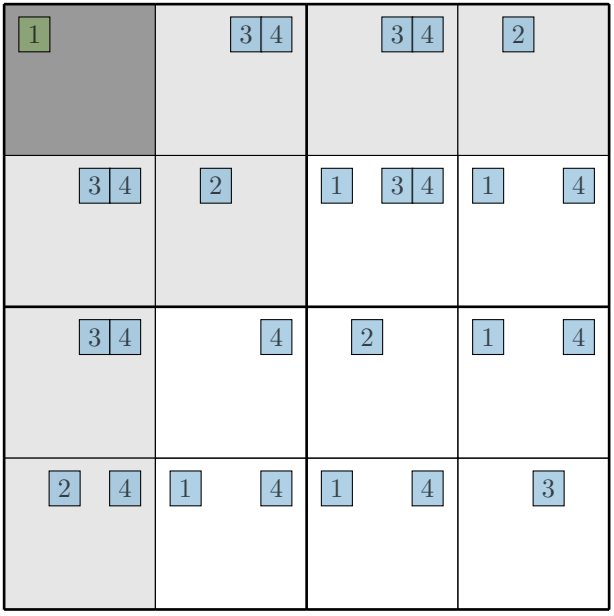


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

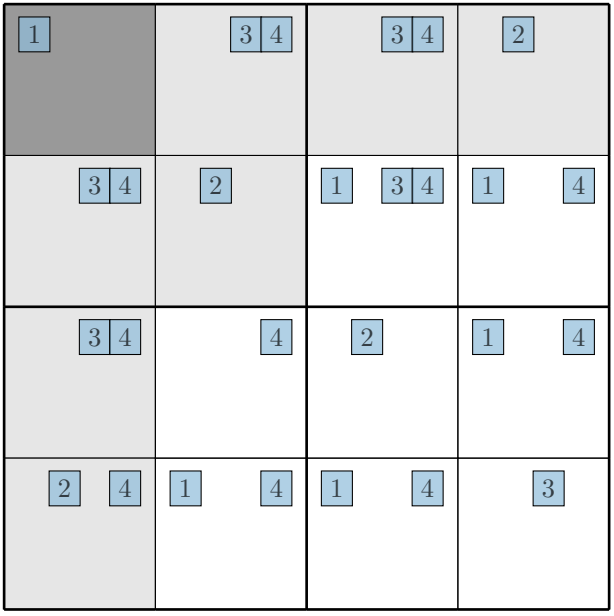


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

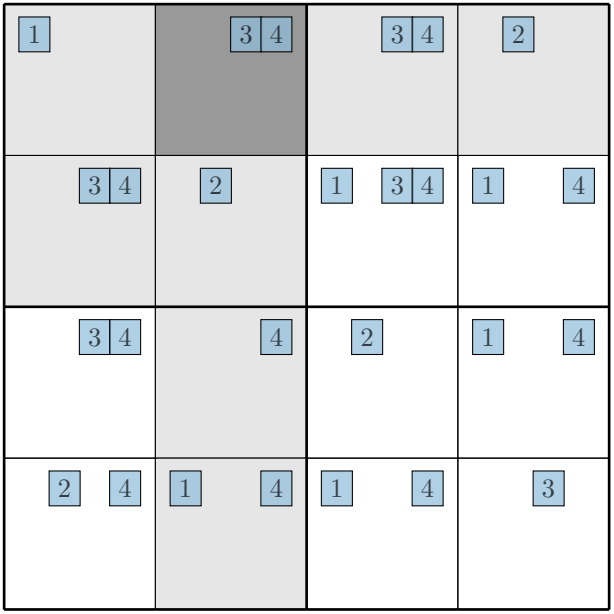


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

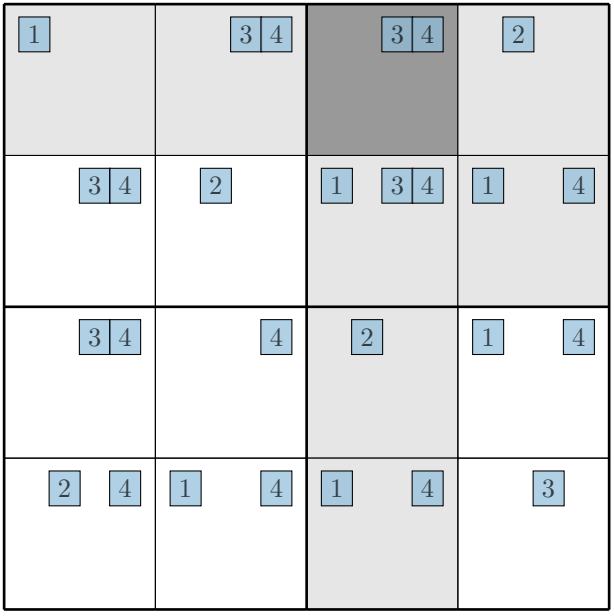


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

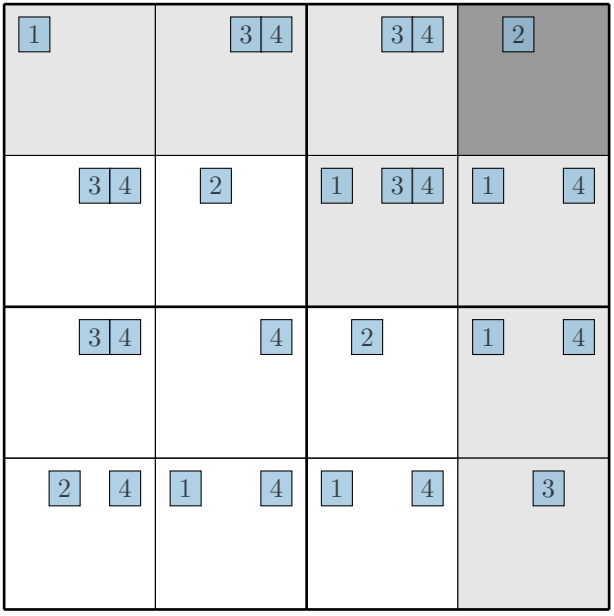


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

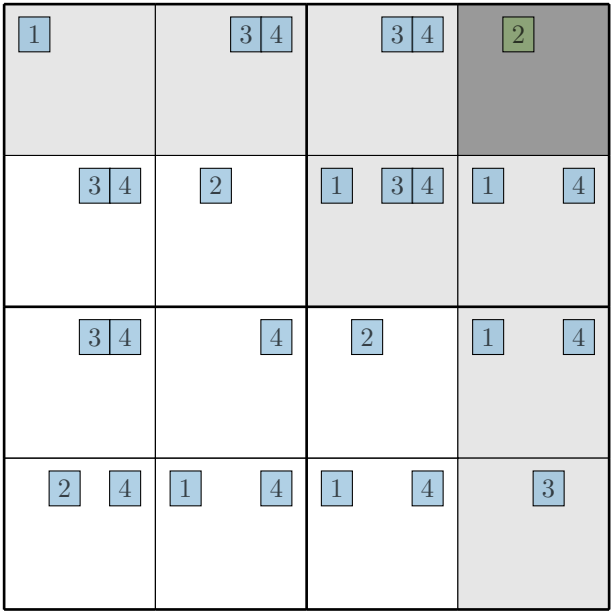


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

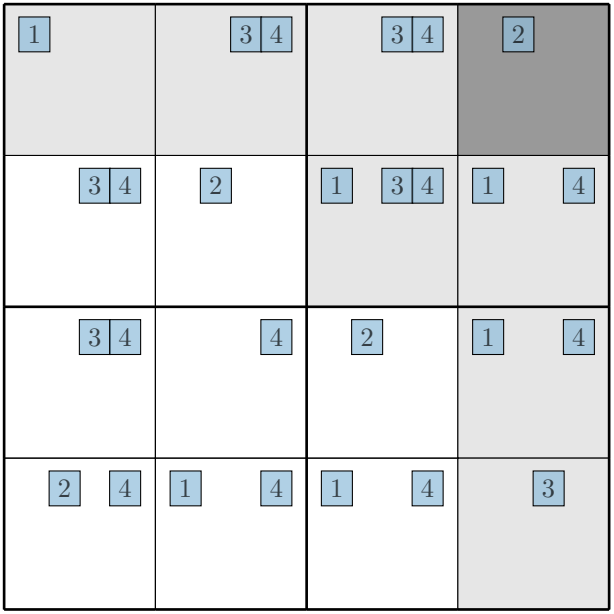


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

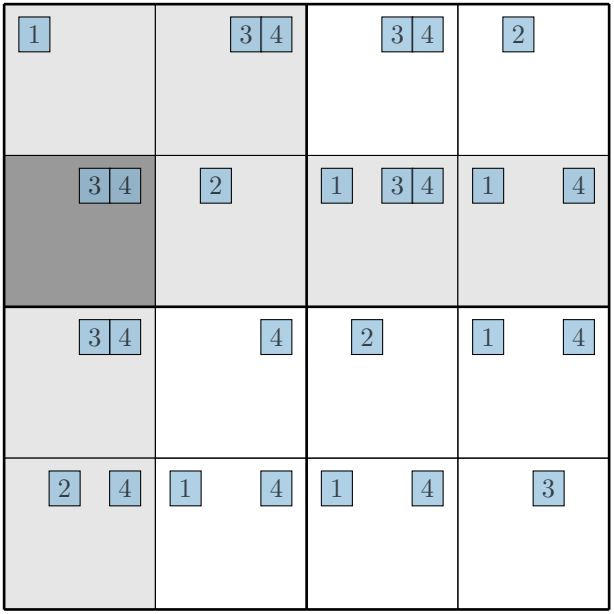


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

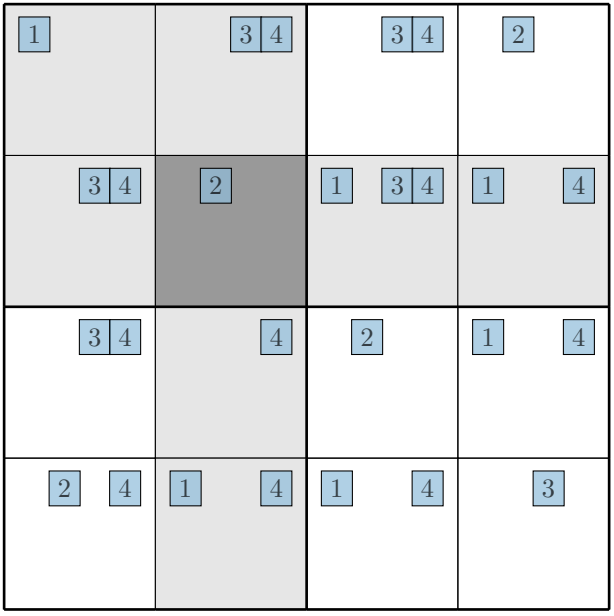


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

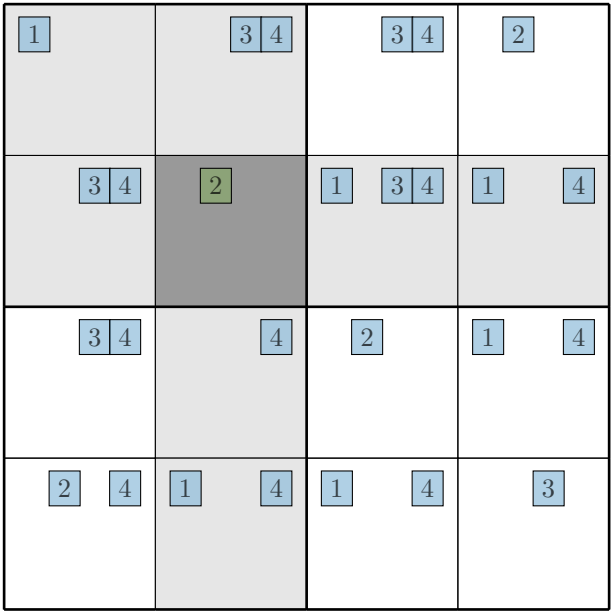


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

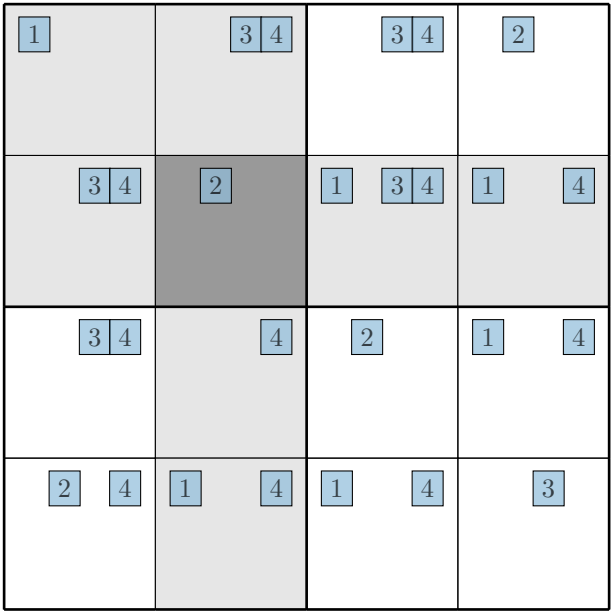


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

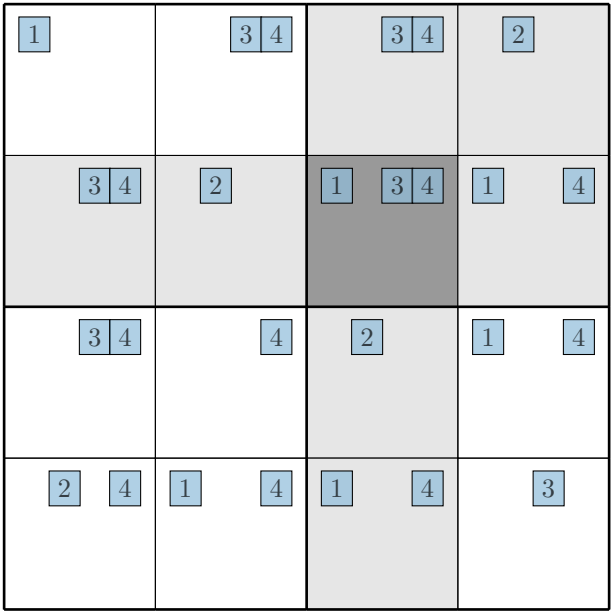


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

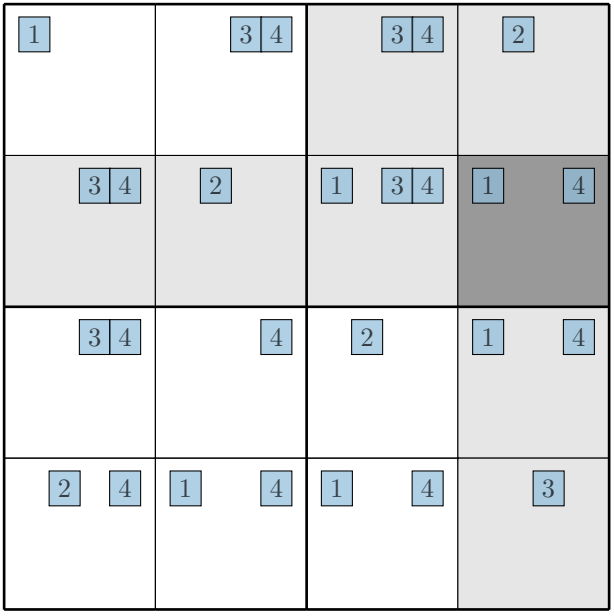


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

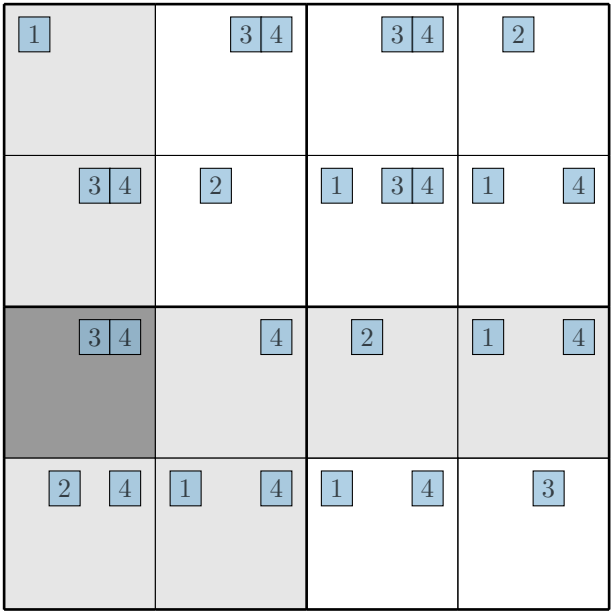


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

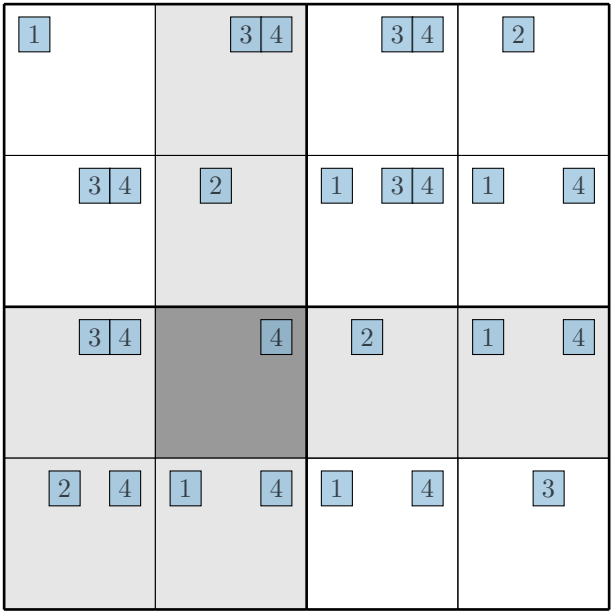


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

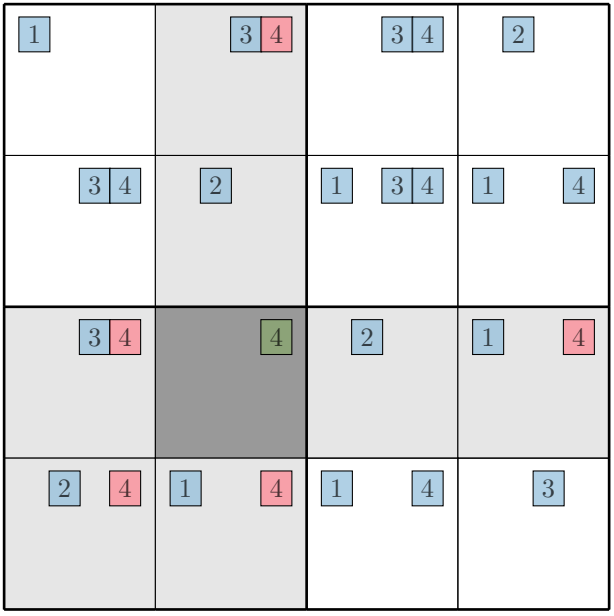


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

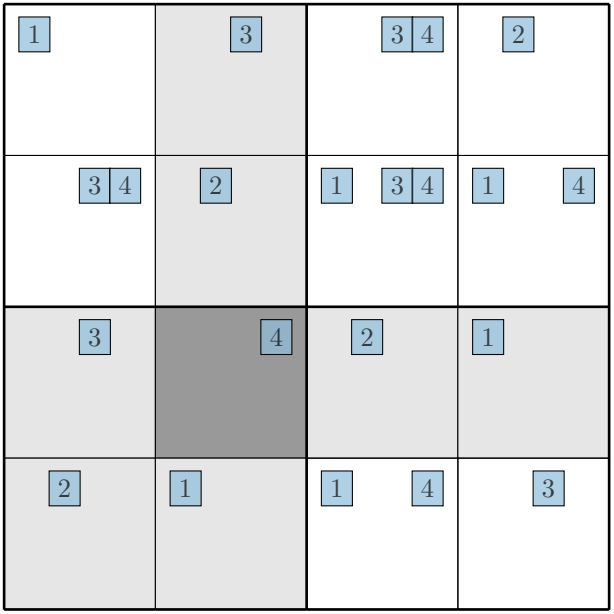


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

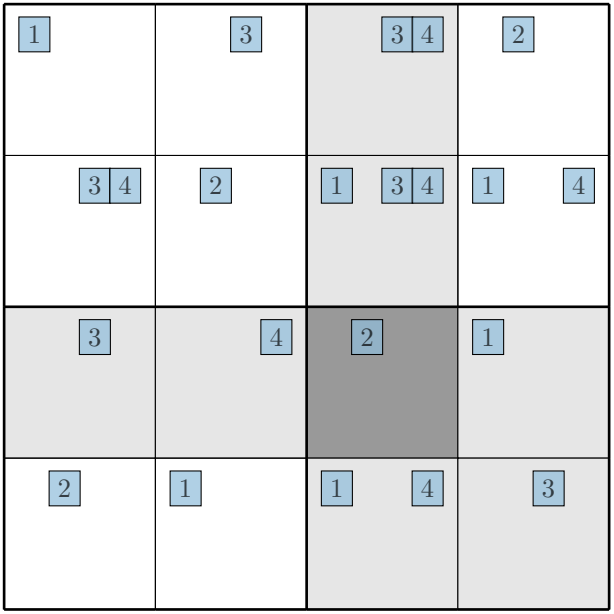


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

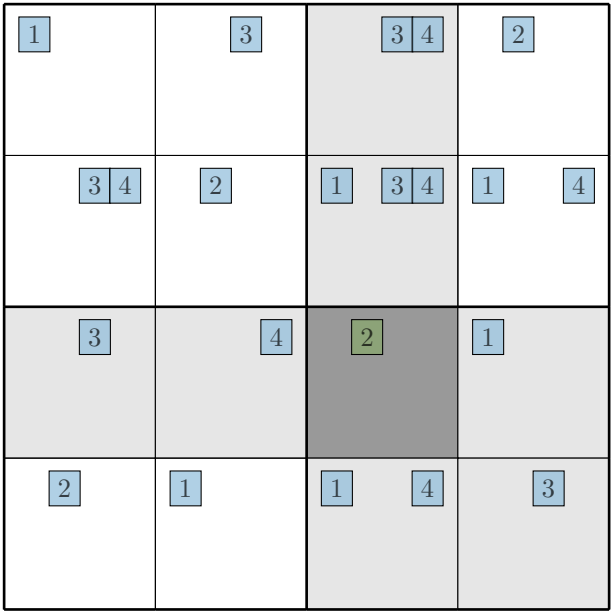


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

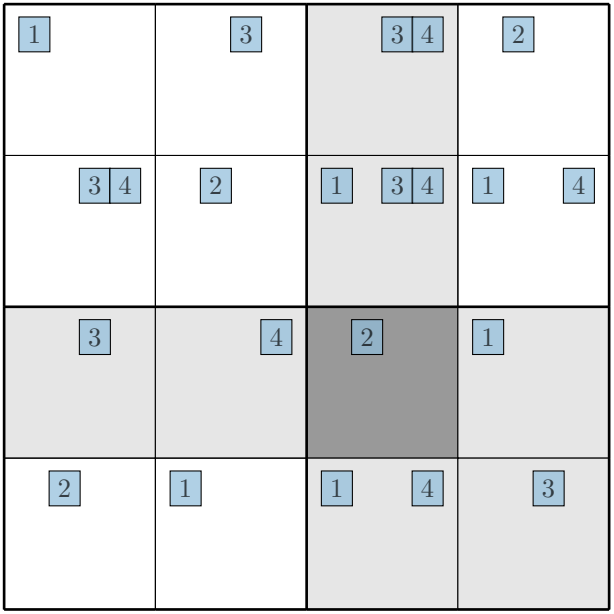


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

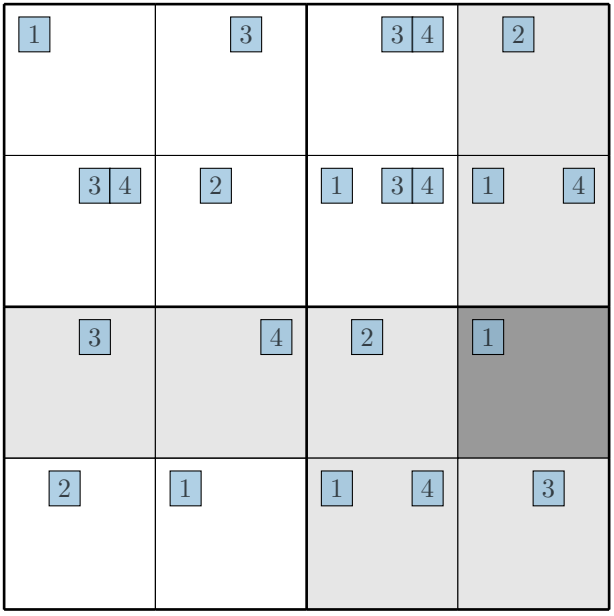


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

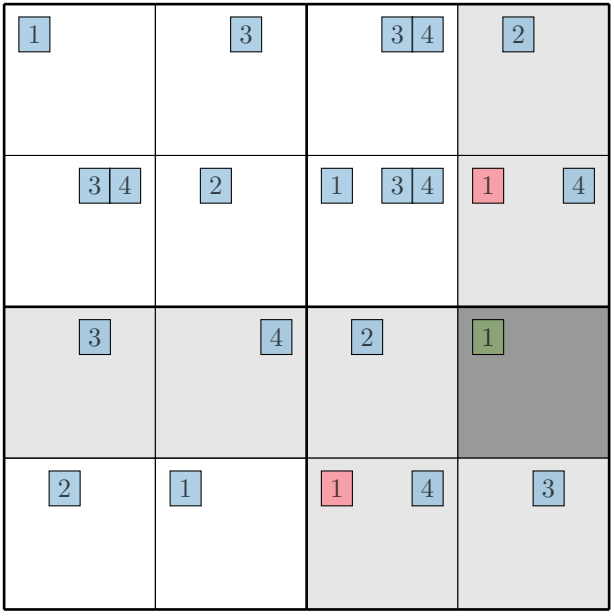


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

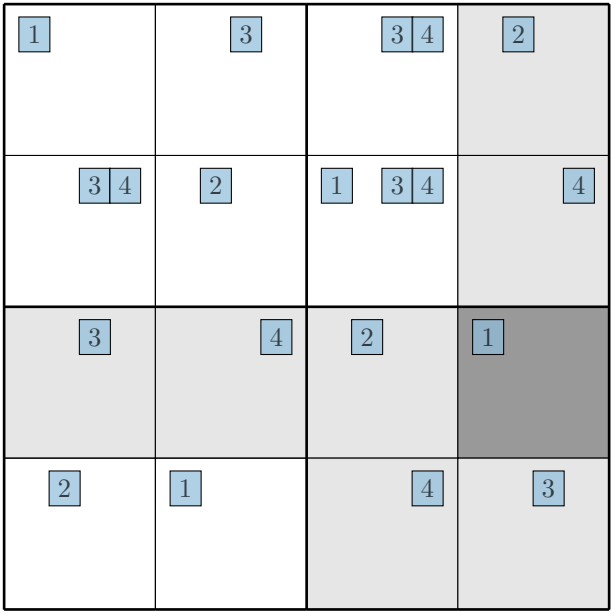


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

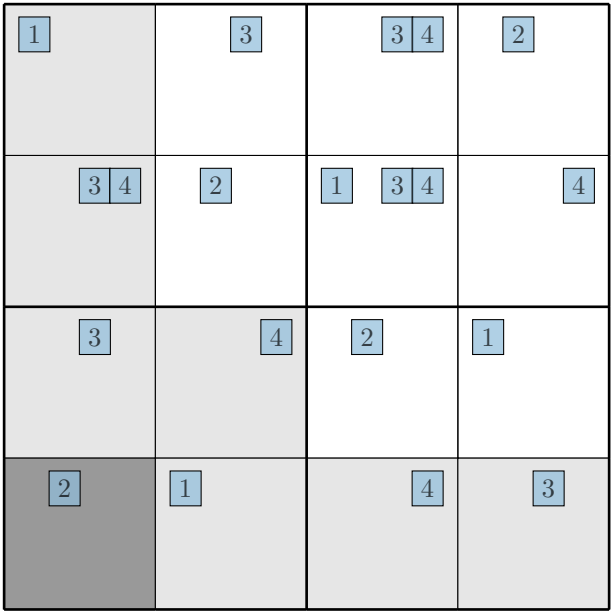


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

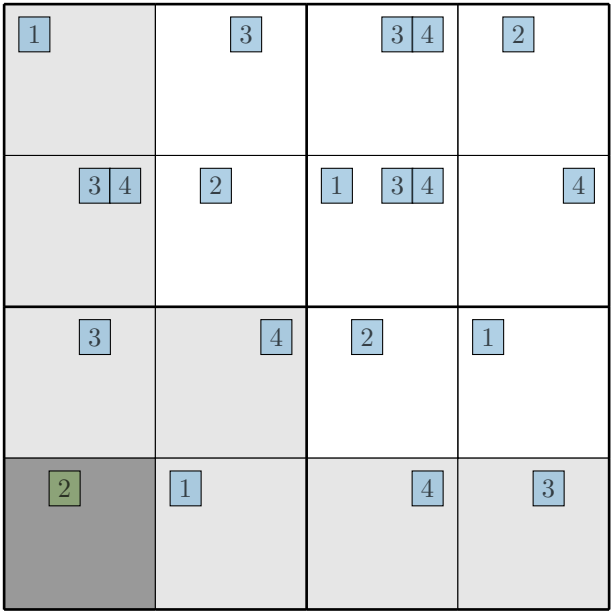


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

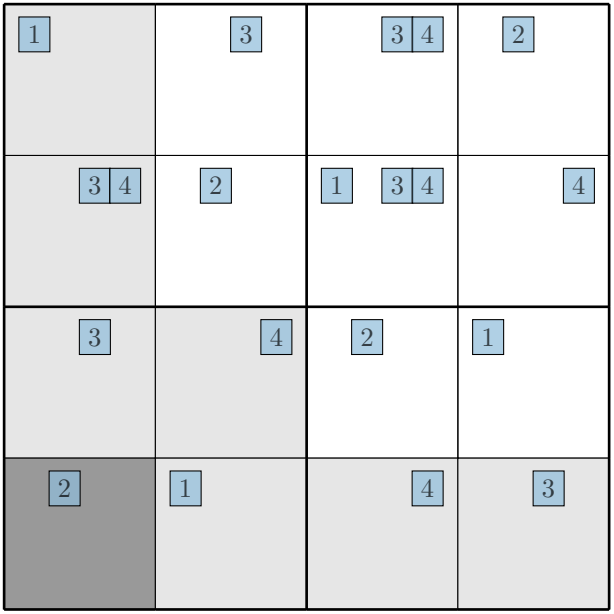


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

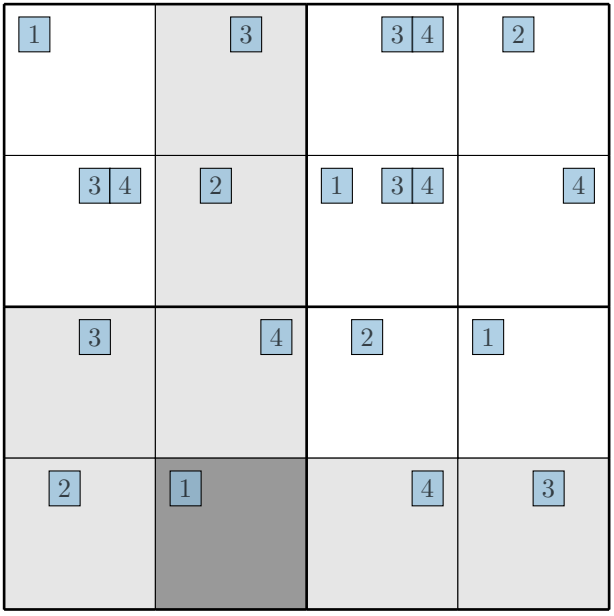


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

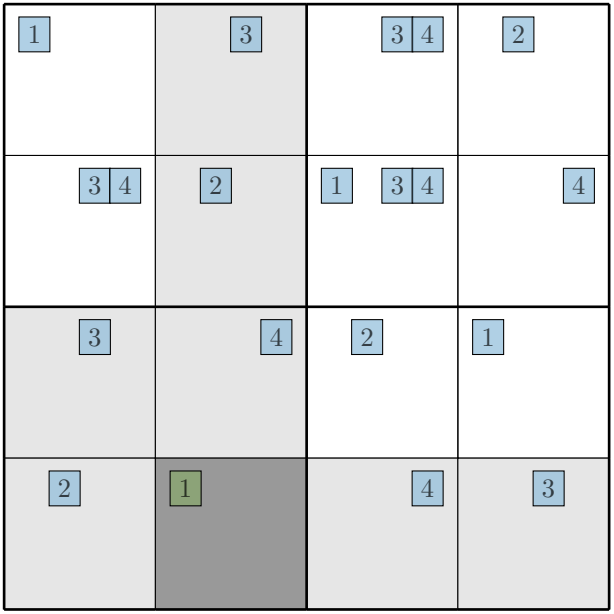


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

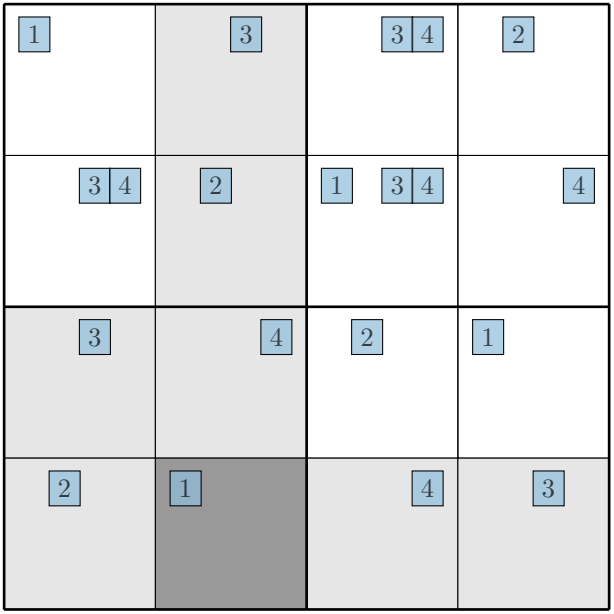


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

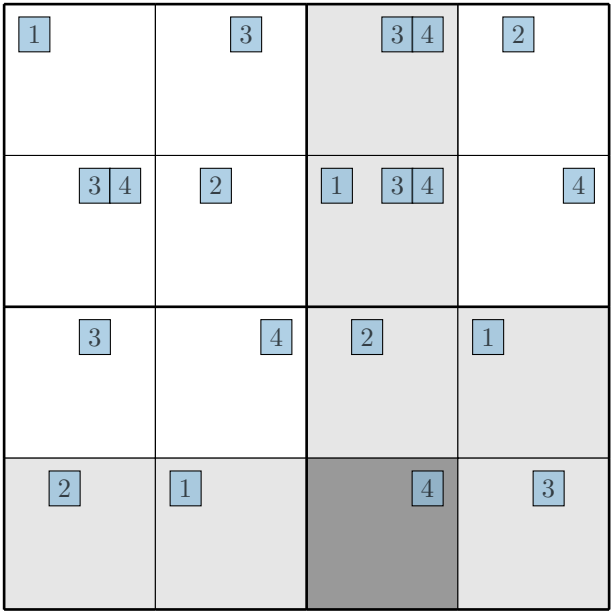


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

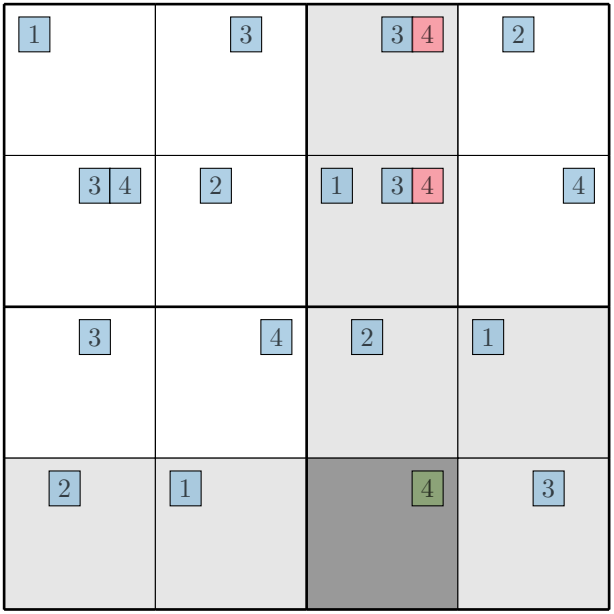


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

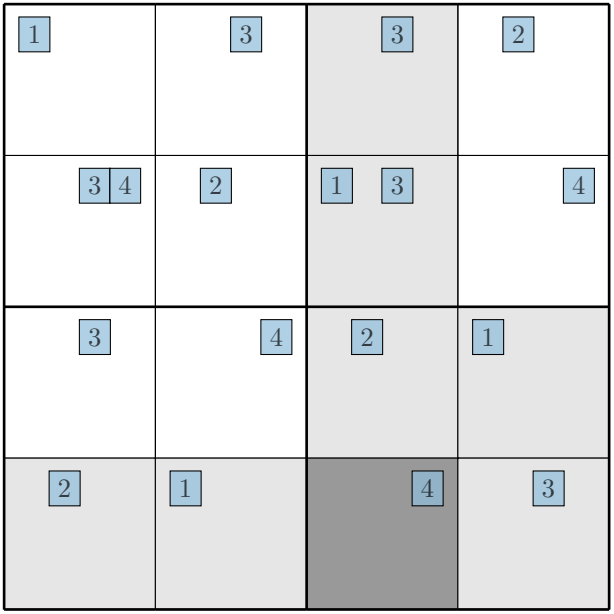


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

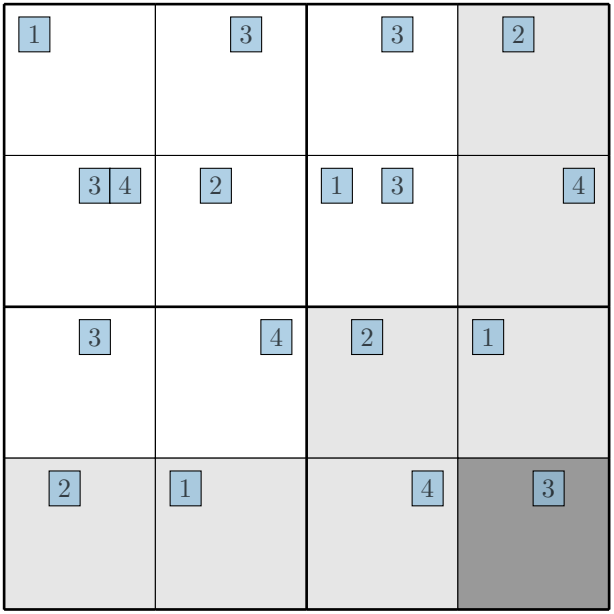


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

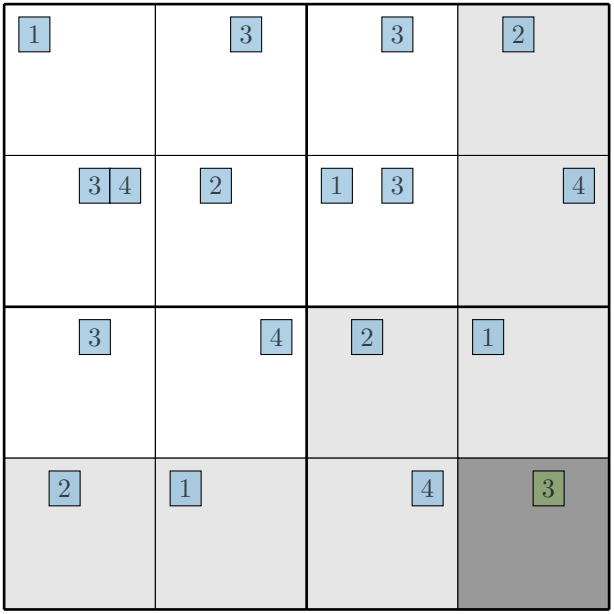


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

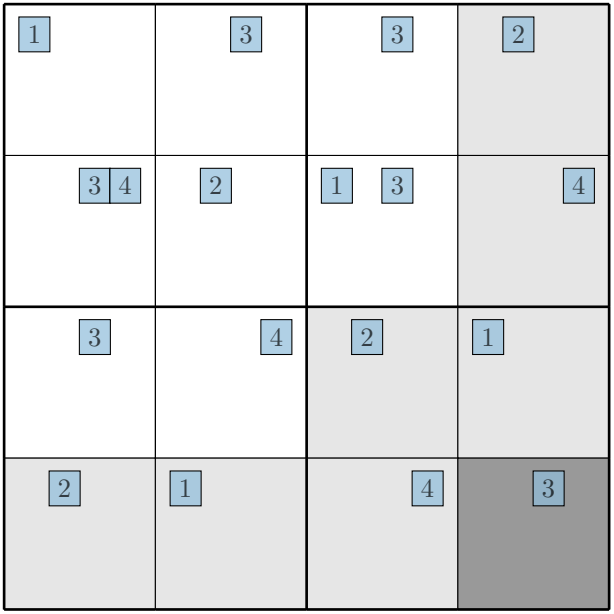


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

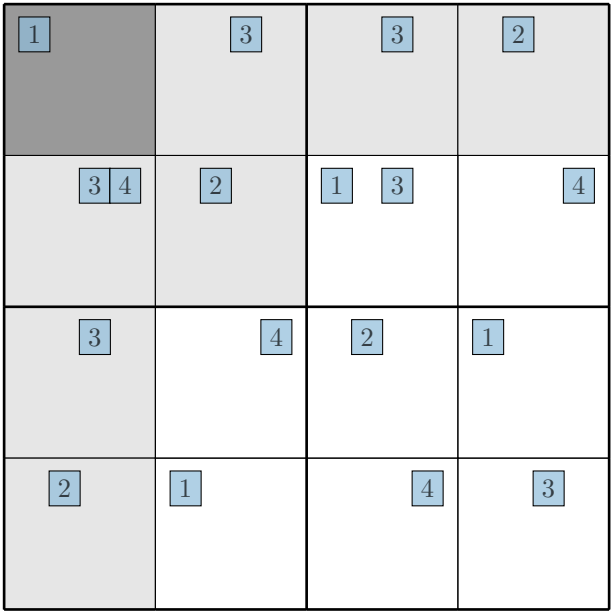


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

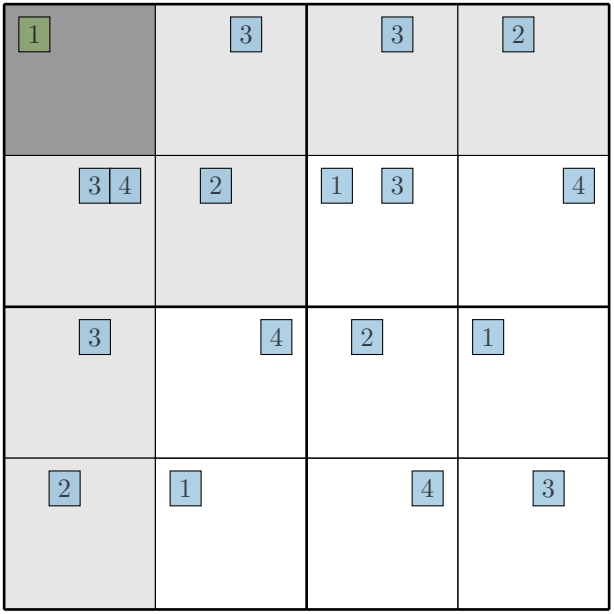


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

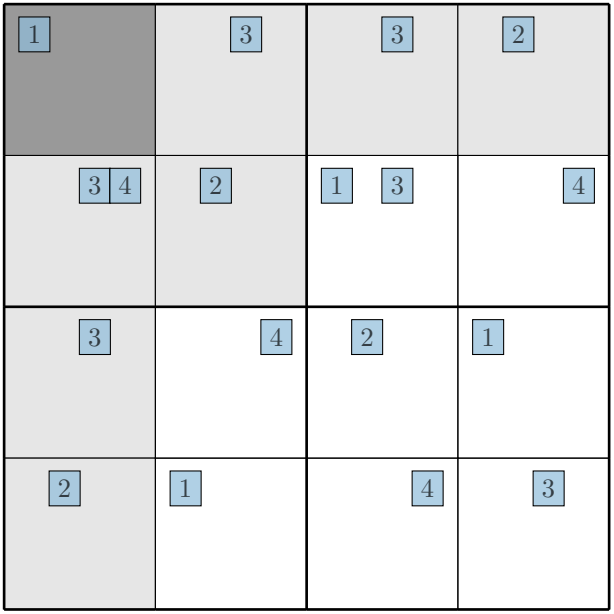


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

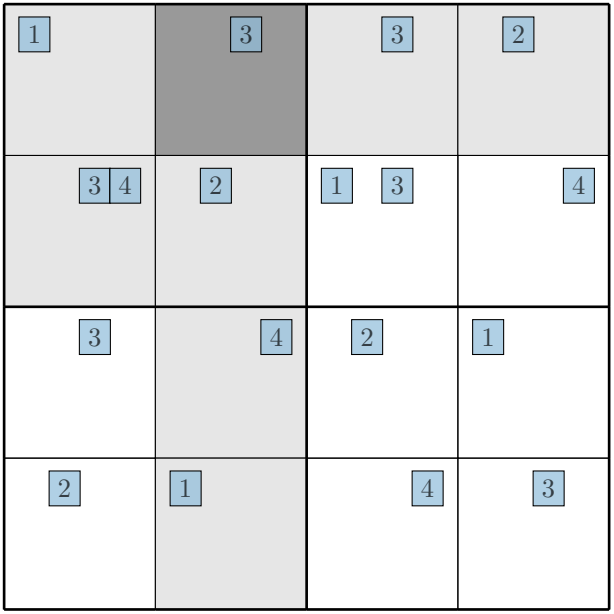


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

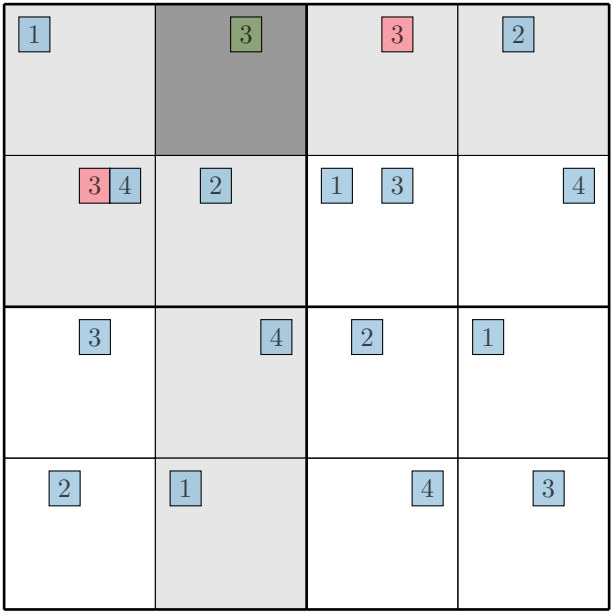


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

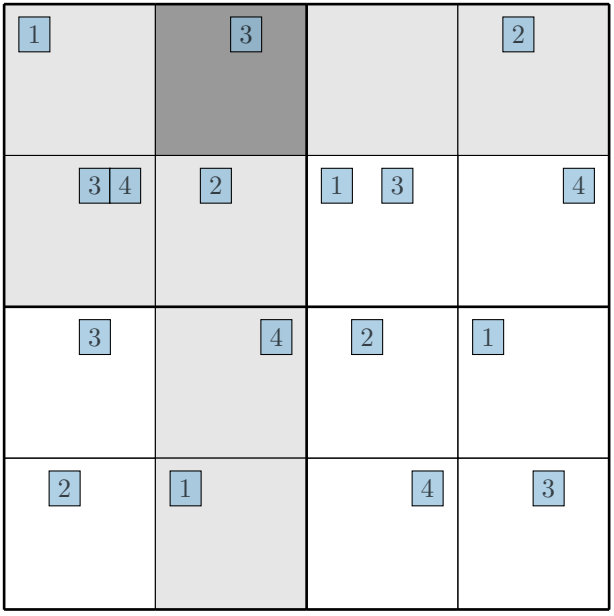


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

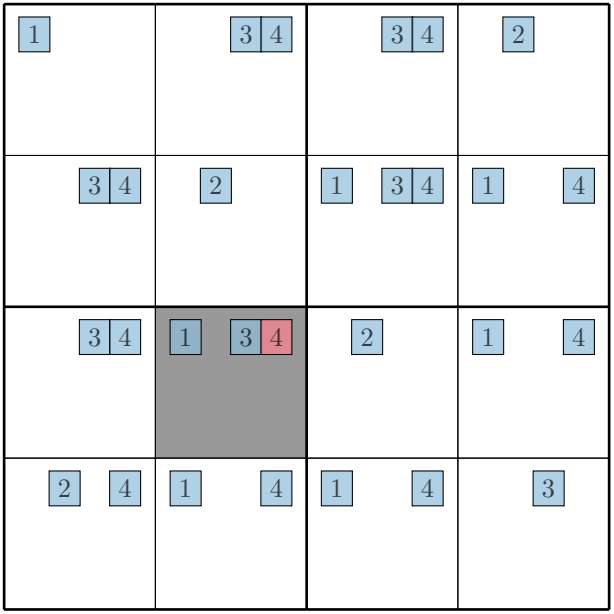


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

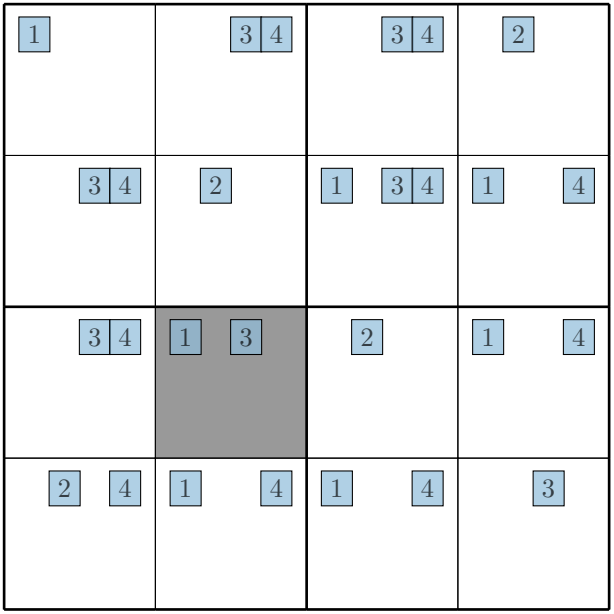


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`backtrack`(`stack`: history stack, `domains`: domains)

- pop (`spot`, `num`, `orig_domains`) from `stack`
- replace `domains` with `orig_domains`
- remove `num` from the domain of `spot`

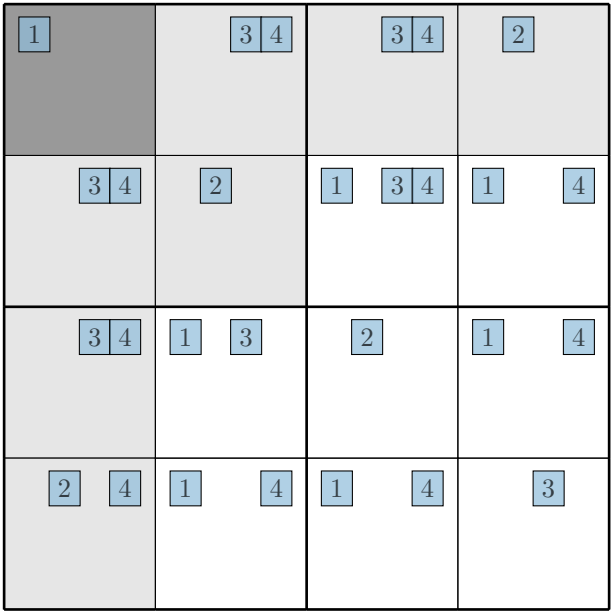


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`backtrack`(`stack`: history stack, `domains`: domains)

- pop (`spot`, `num`, `orig_domains`) from `stack`
- replace `domains` with `orig_domains`
- remove `num` from the domain of `spot`

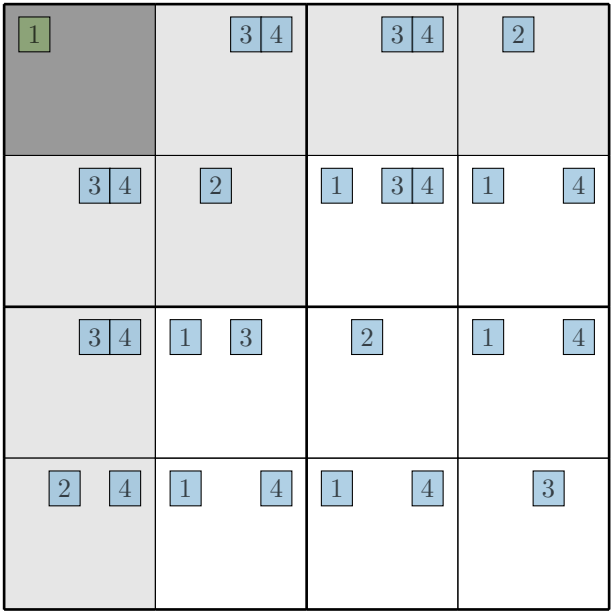


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

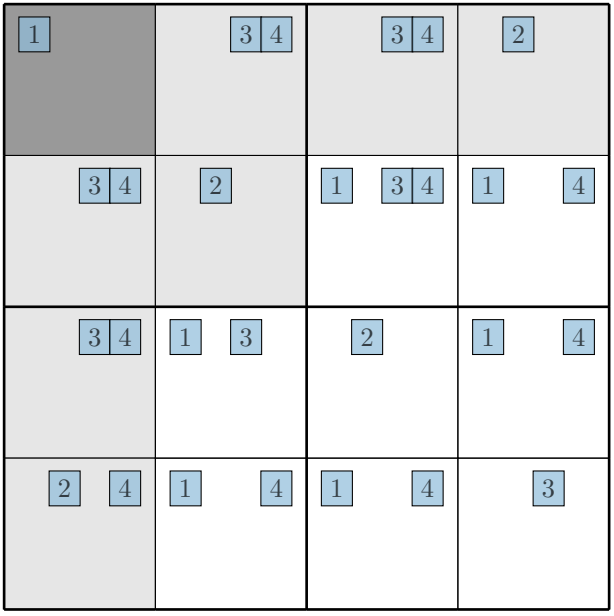


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

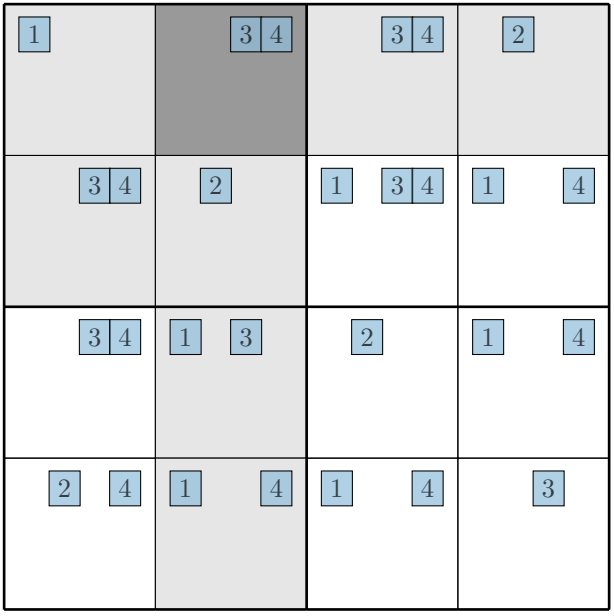


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

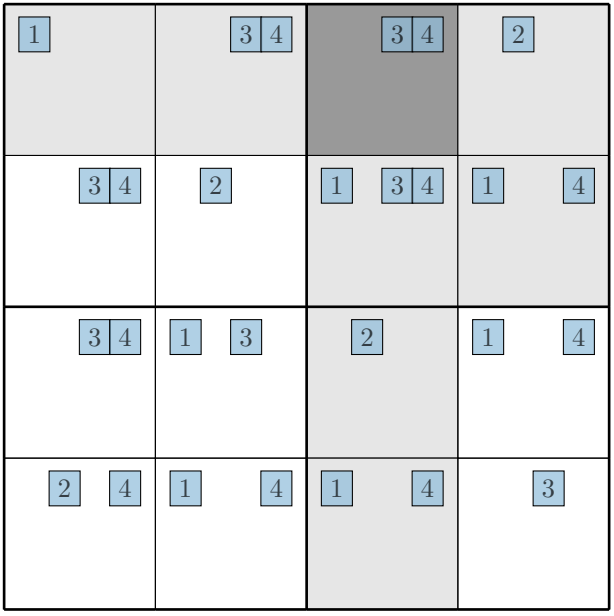


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

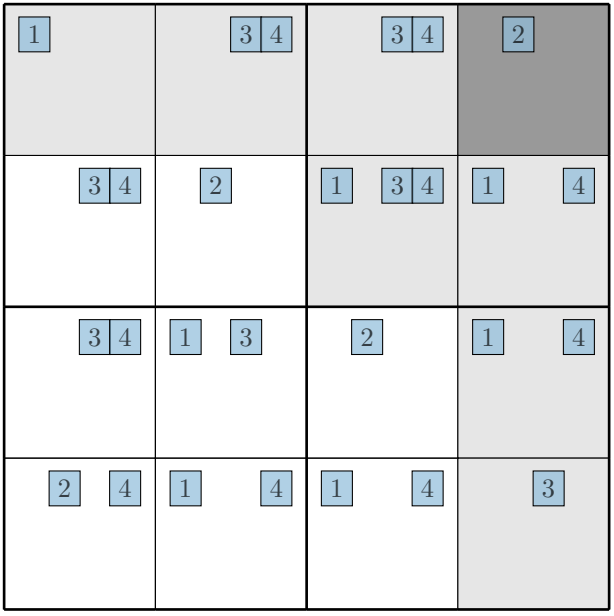


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

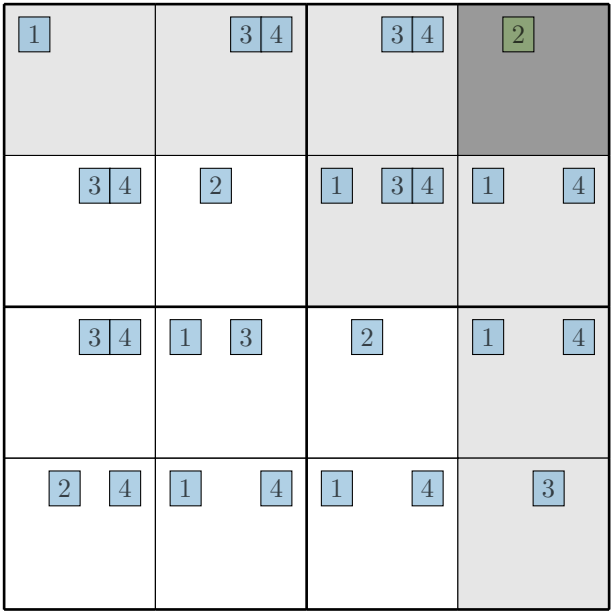


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

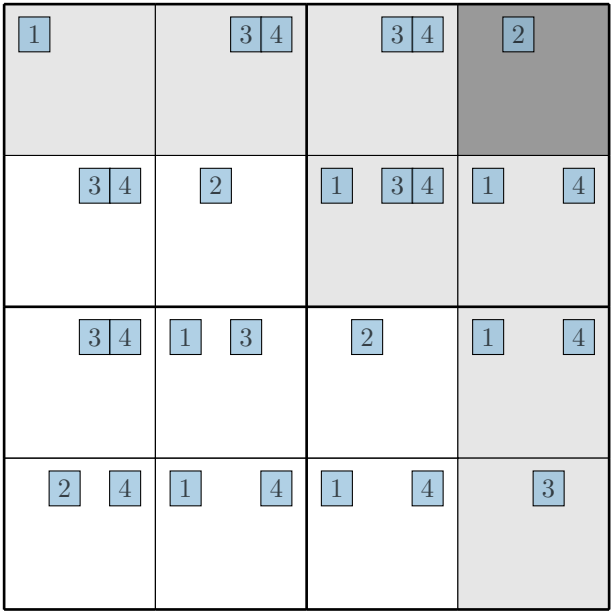


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

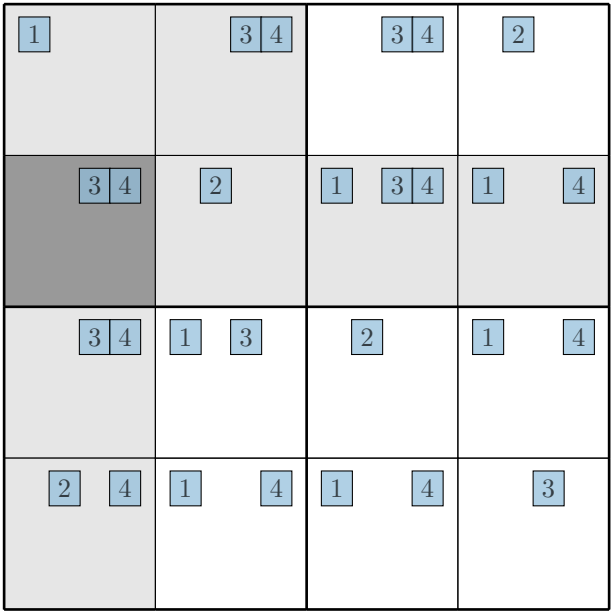


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

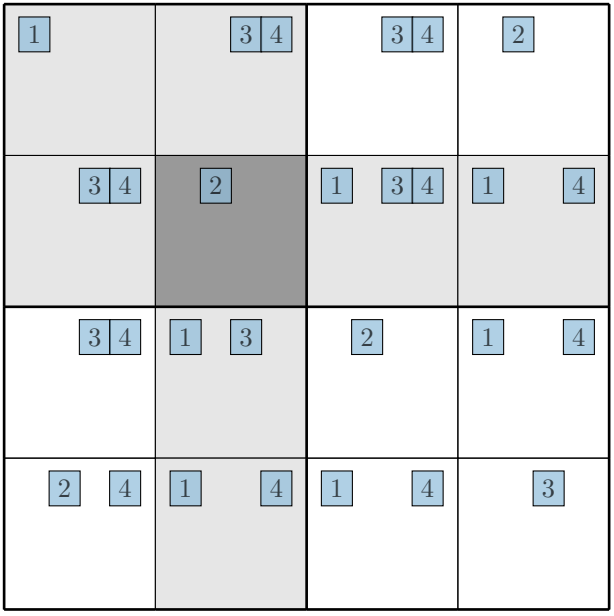


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

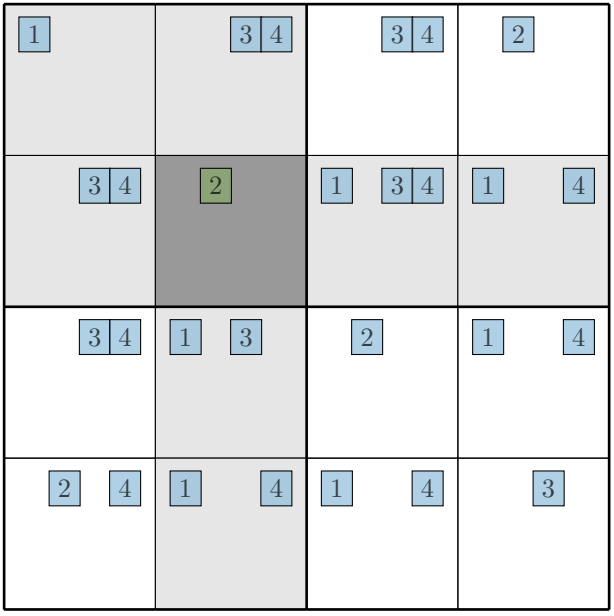


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

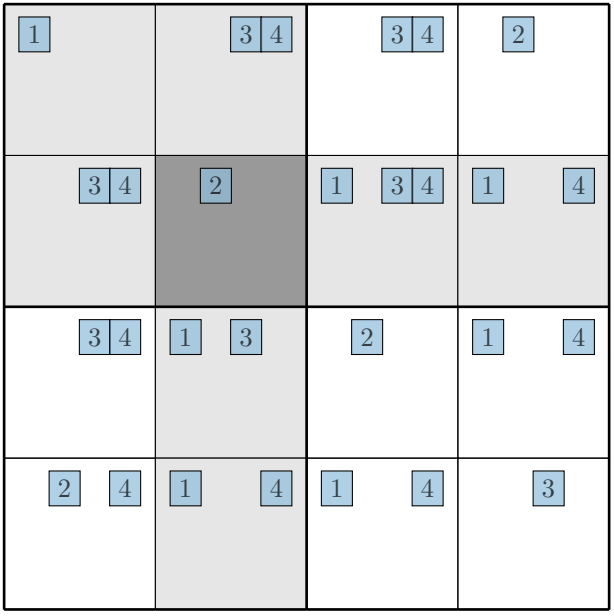


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

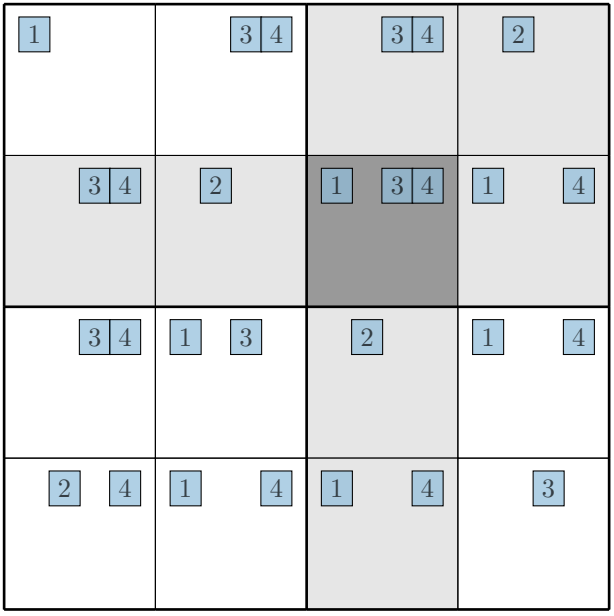


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

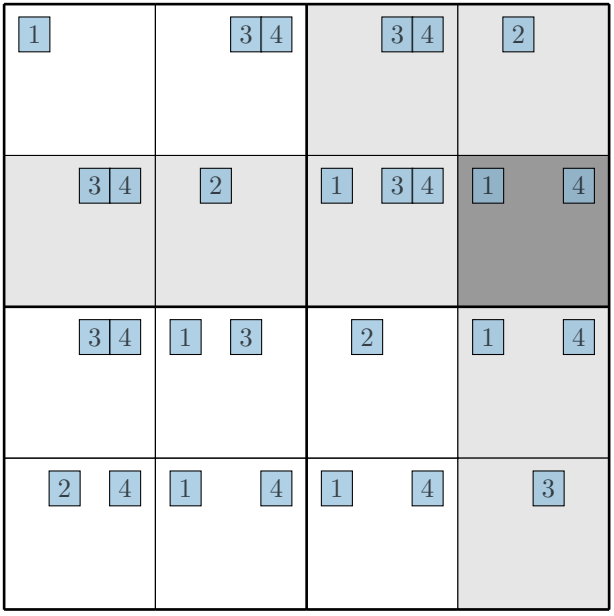


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

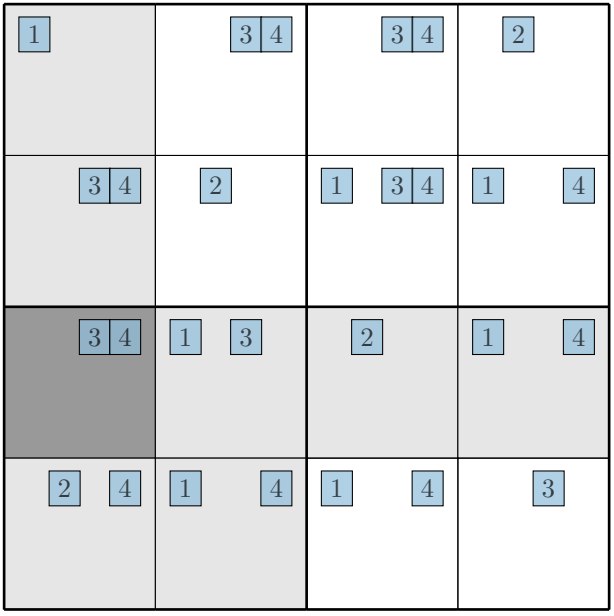


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

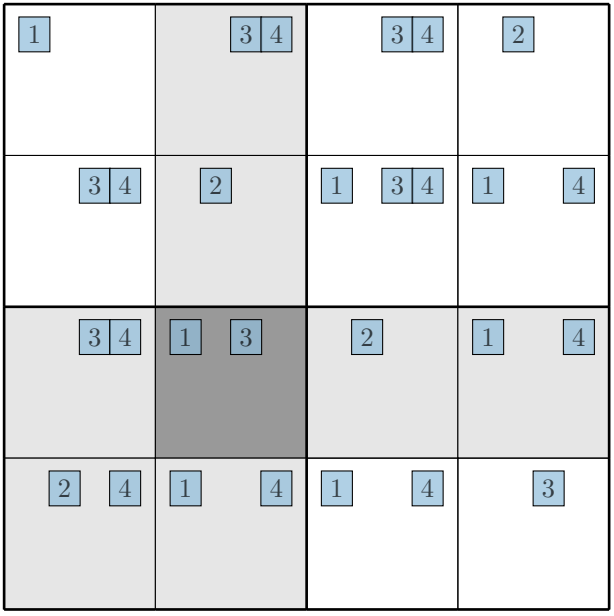


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

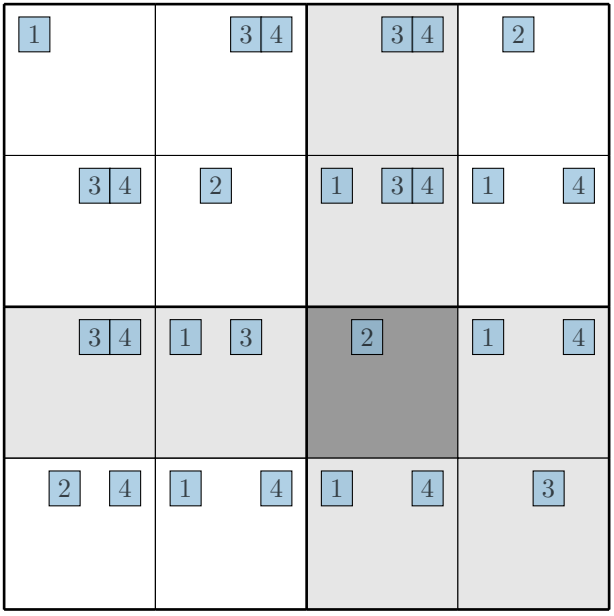


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

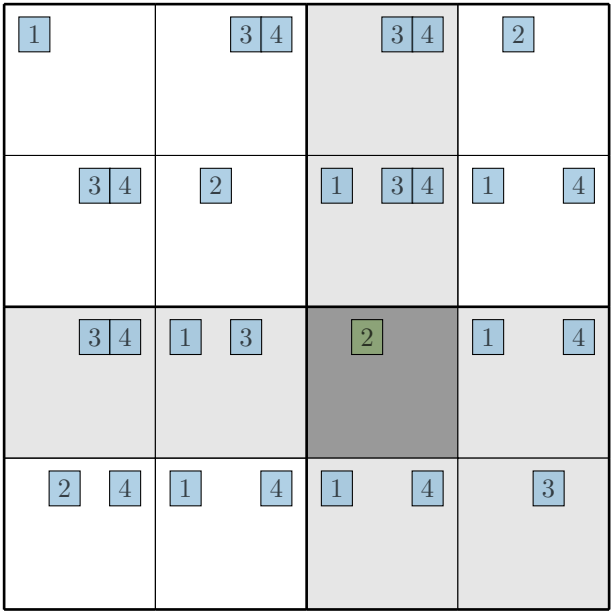


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

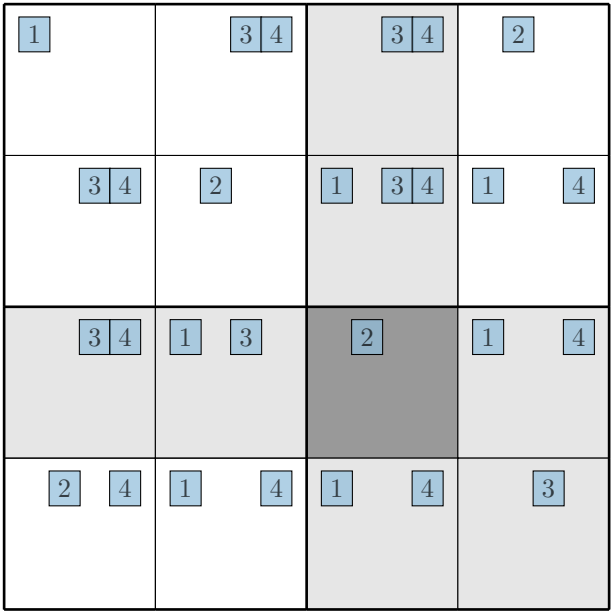


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

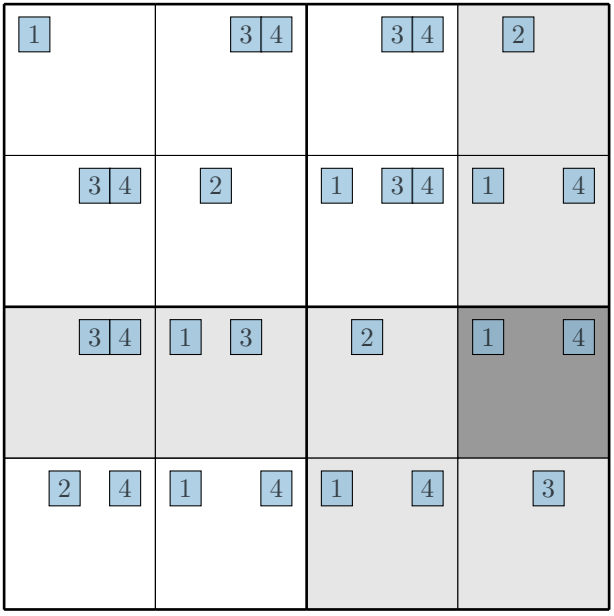


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

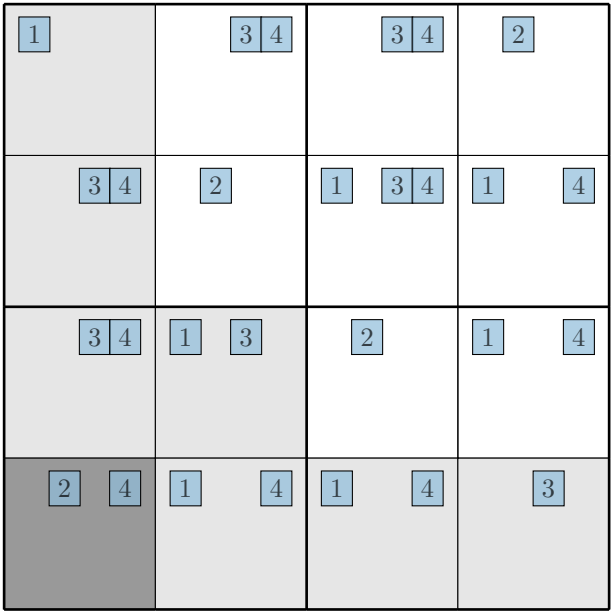


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

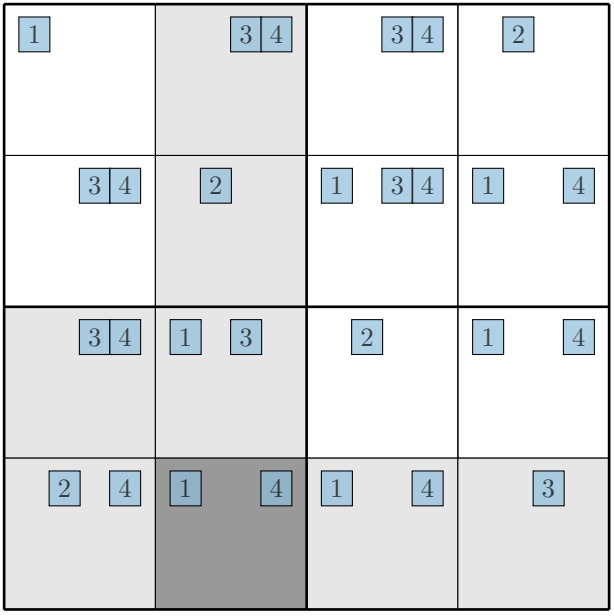


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

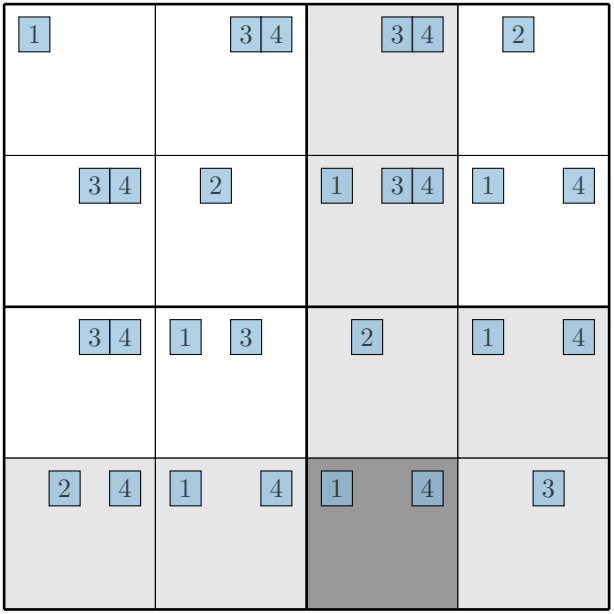


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

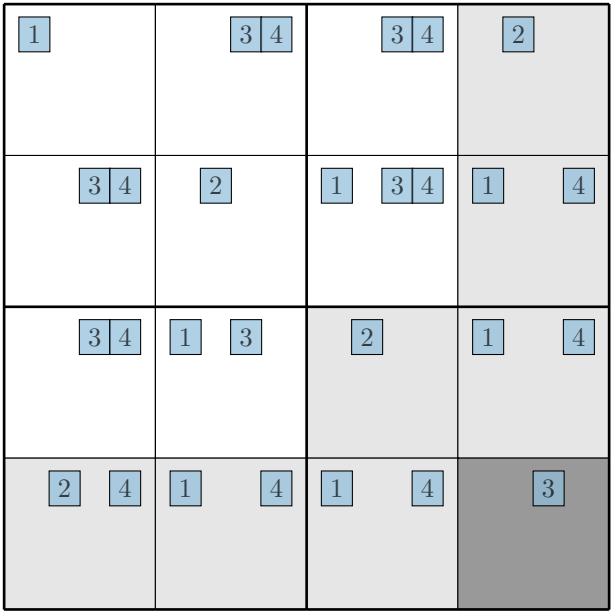


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

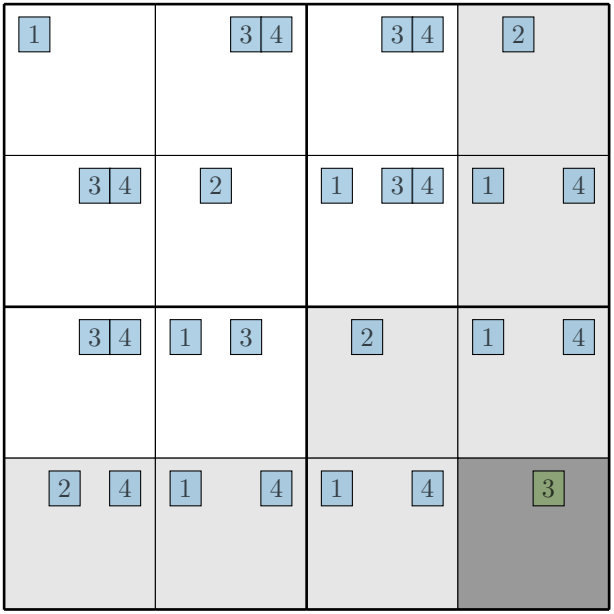


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

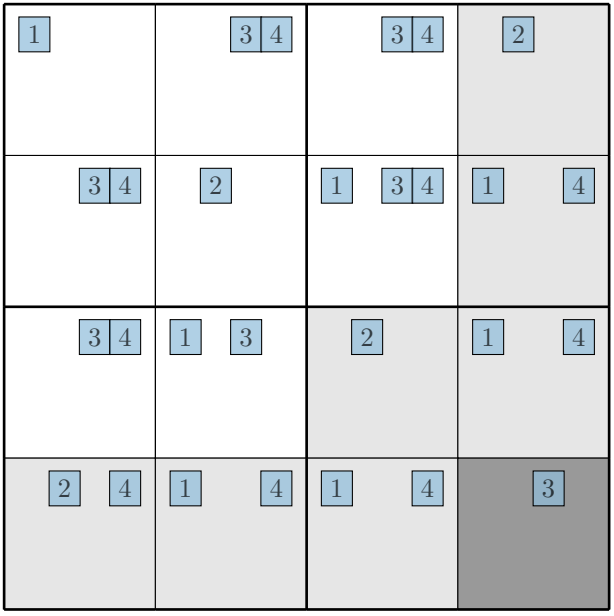


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

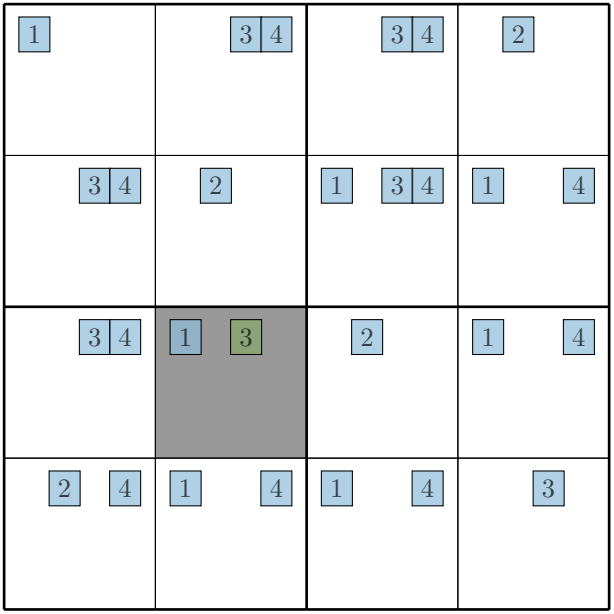


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

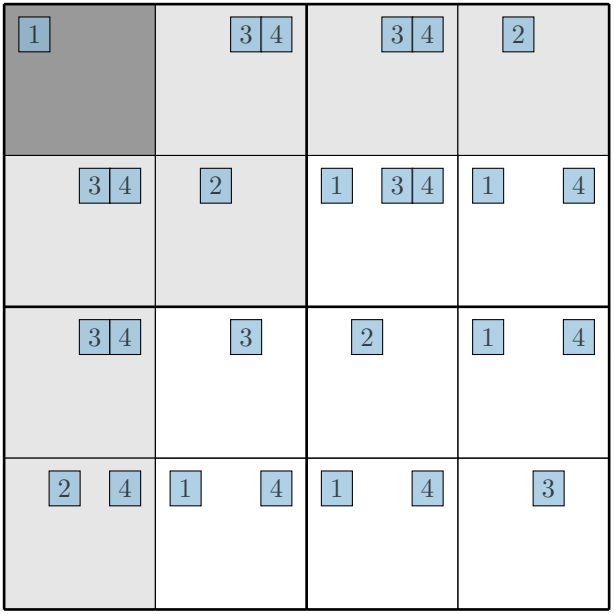


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`search(domains`: tile domains)

- `spot` = an unassigned spot (according to `domains`)
- `num` = a possible num for `spot` (according to `domains`)
- restrict the domain of `spot` to `num`.

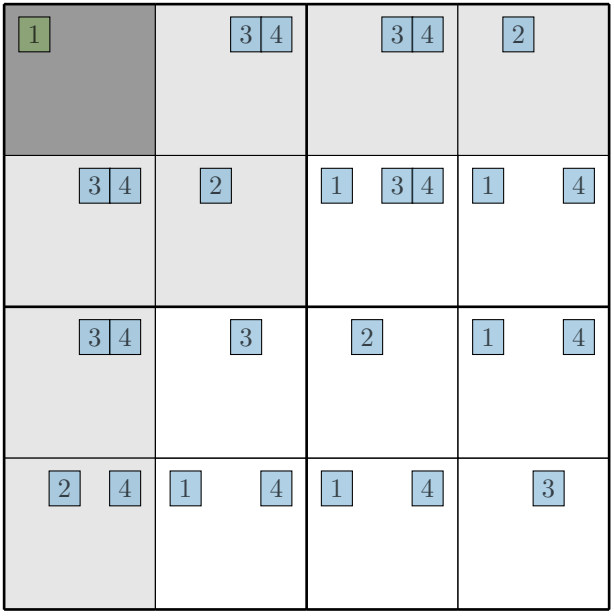


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

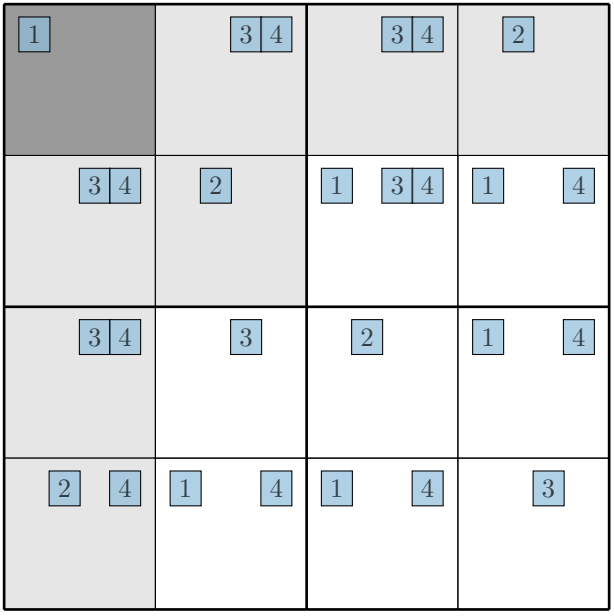


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

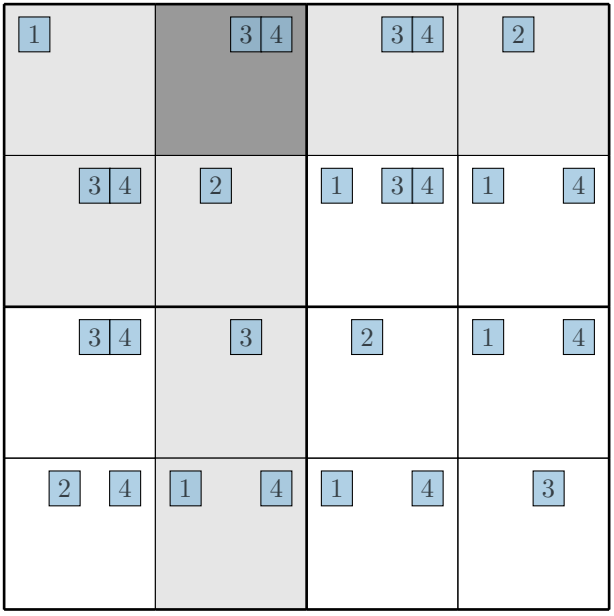


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

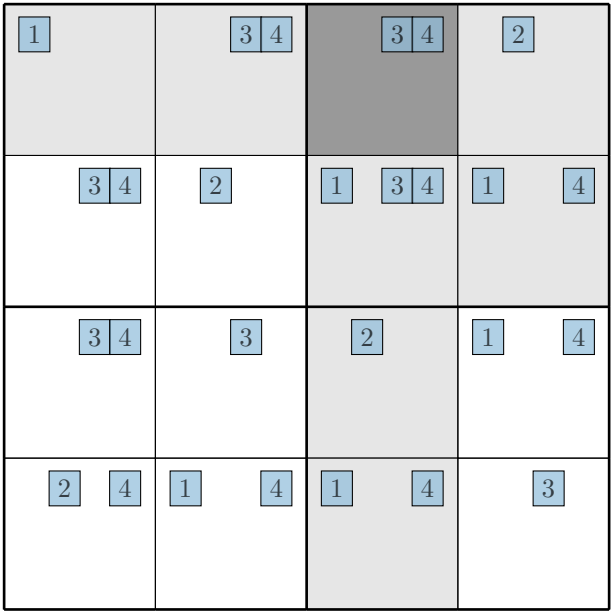


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

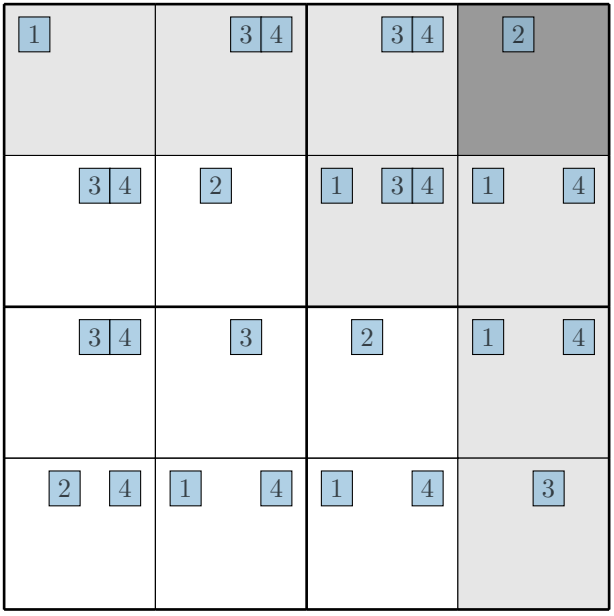


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

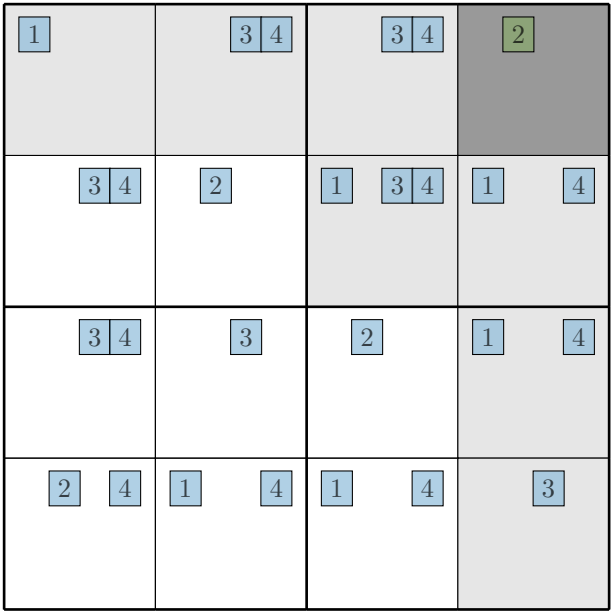


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

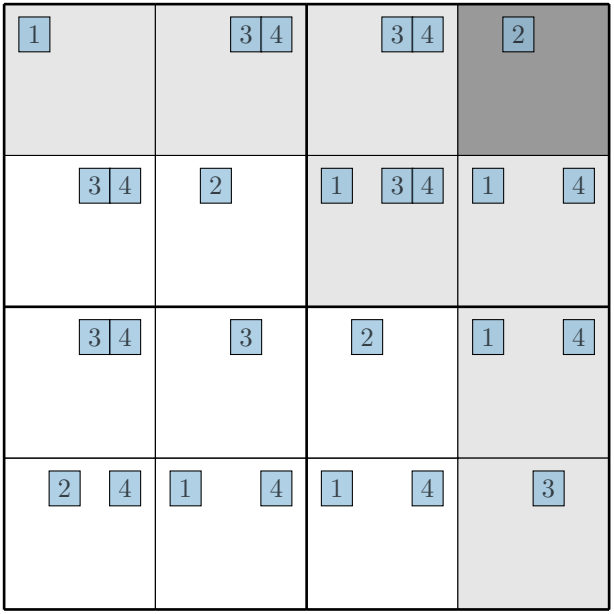


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

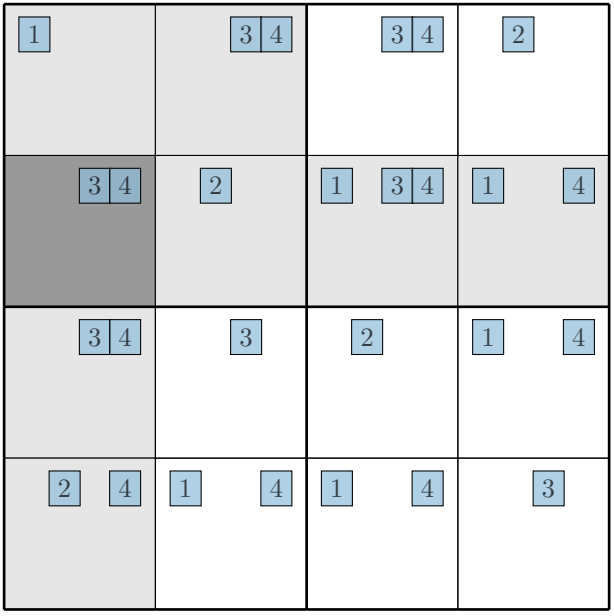


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

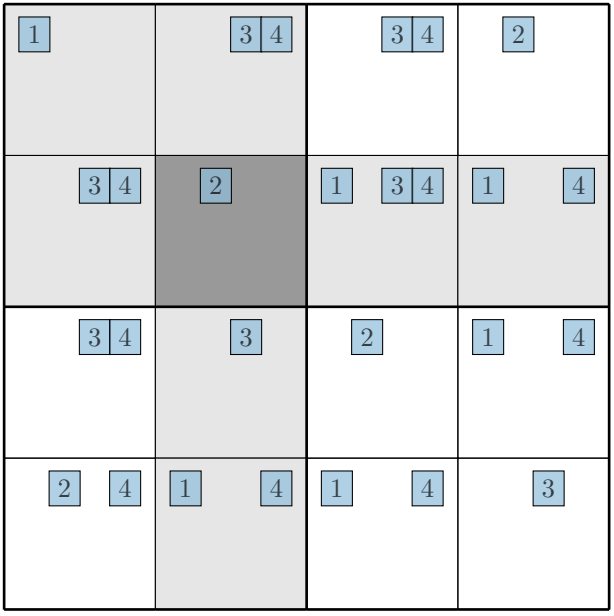


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

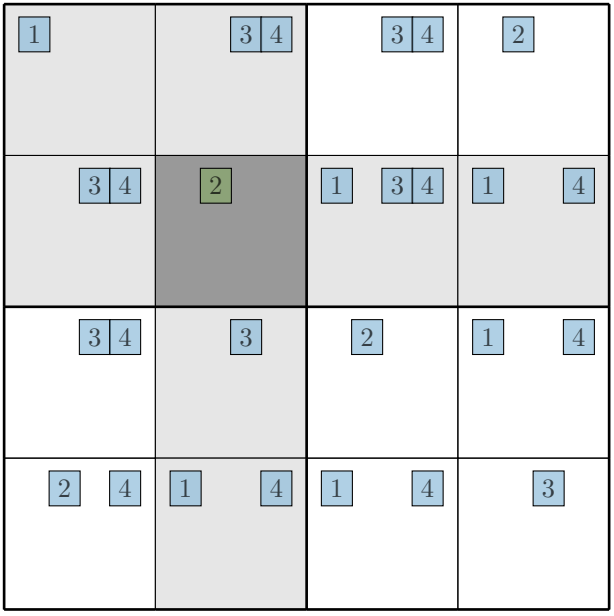


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

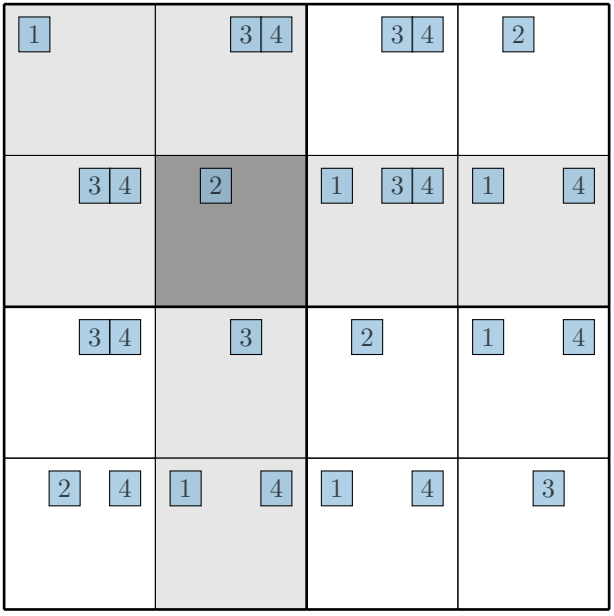


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

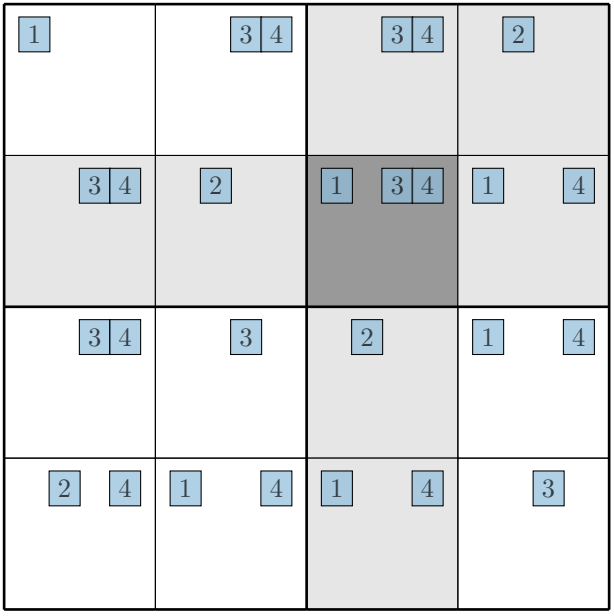


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

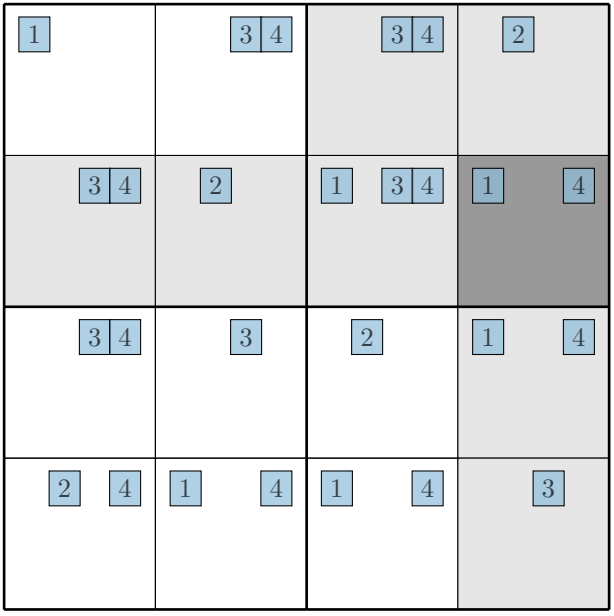


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

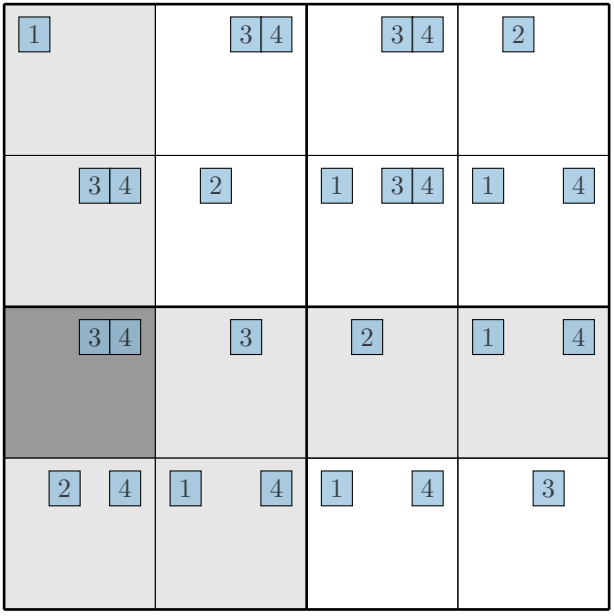


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

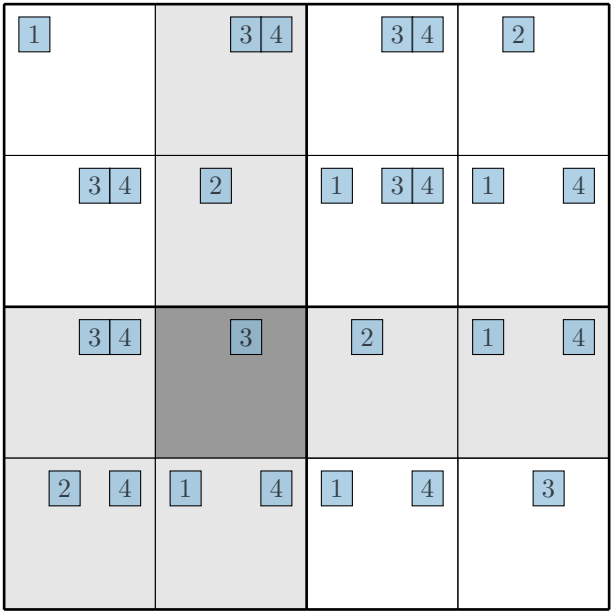


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

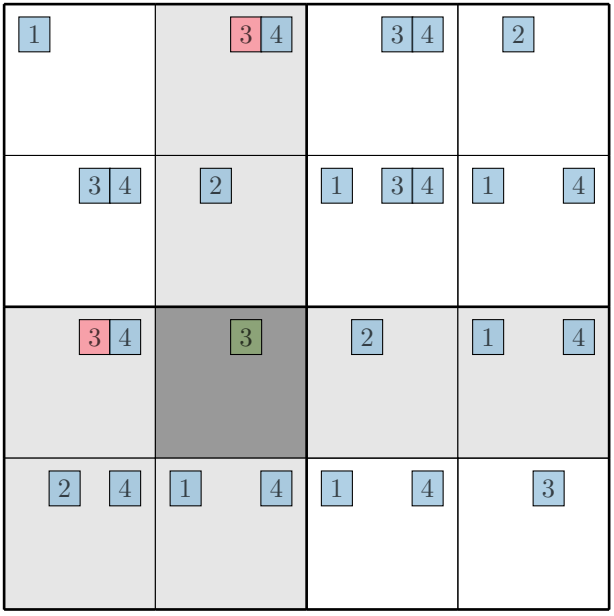


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

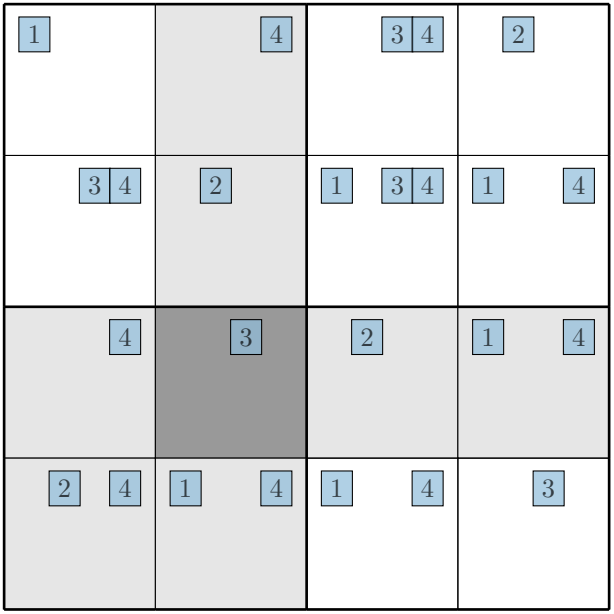


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

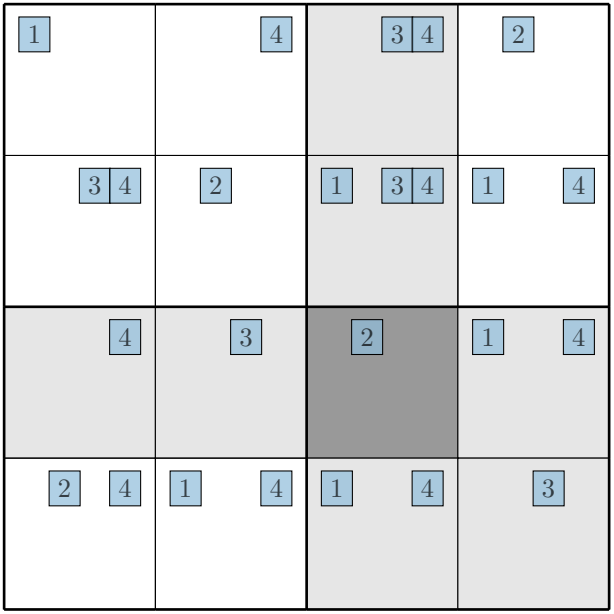


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

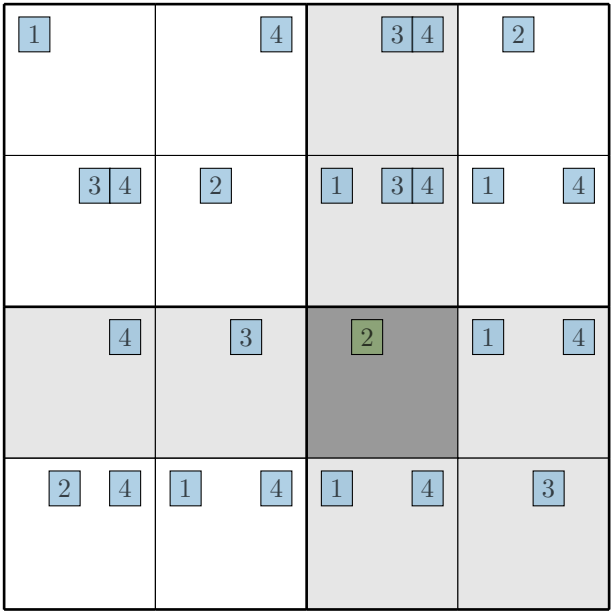


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

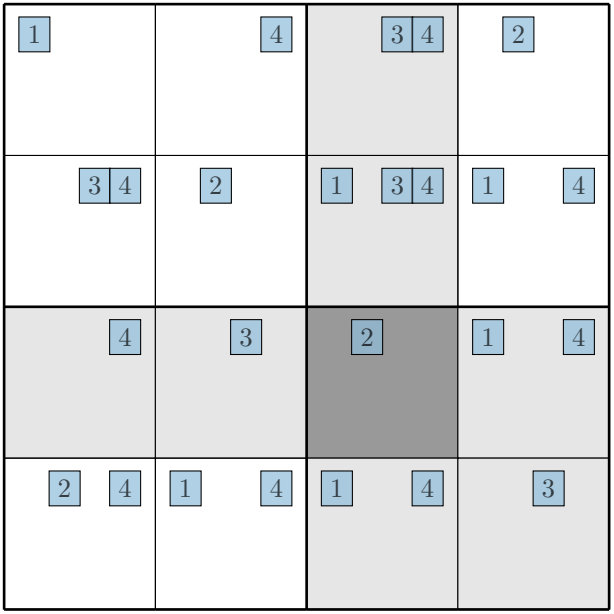


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

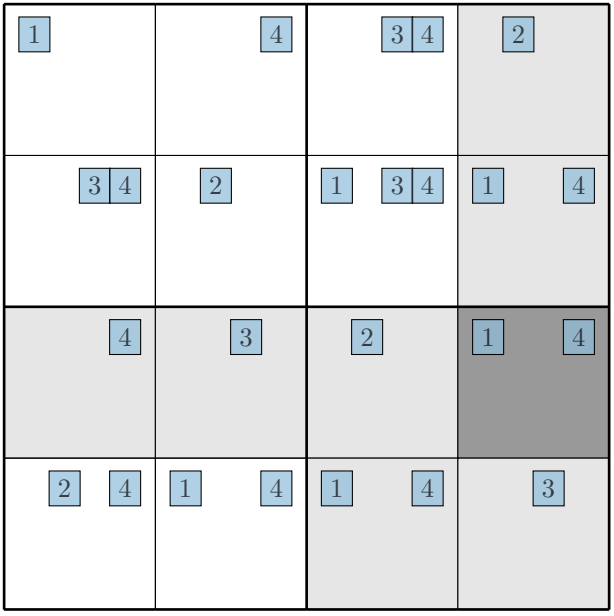


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

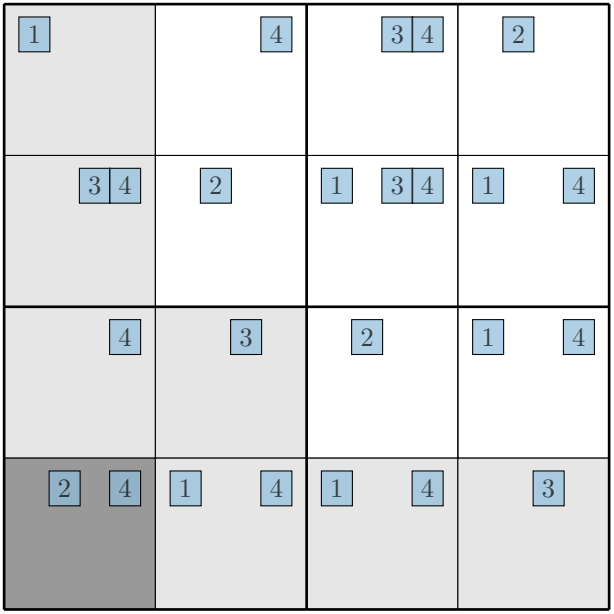


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

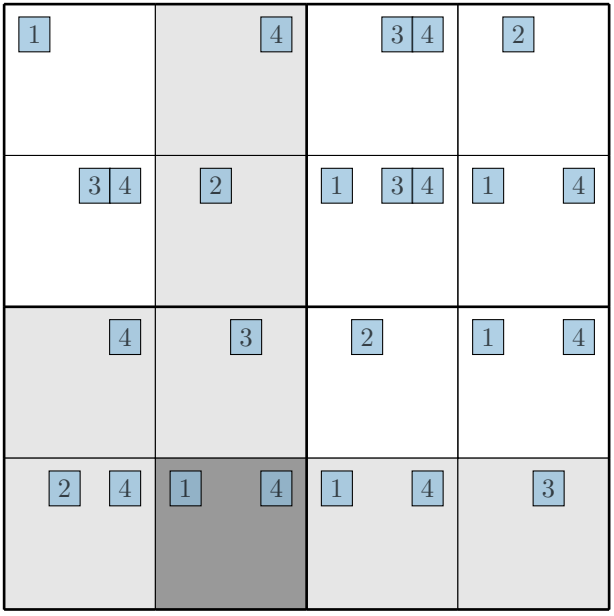


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

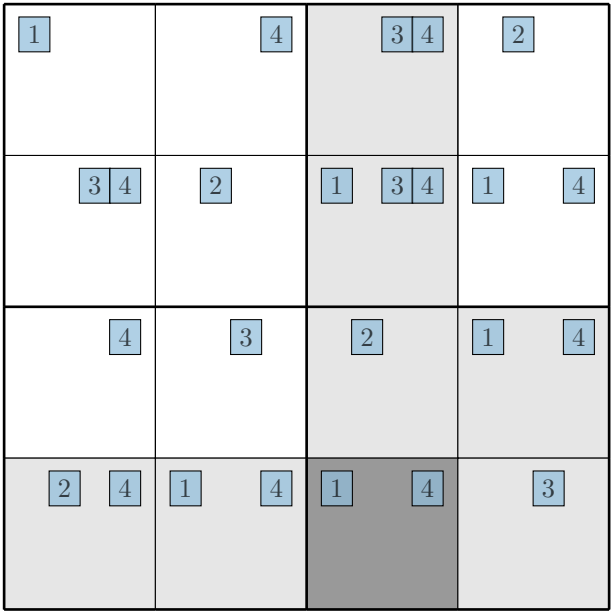


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

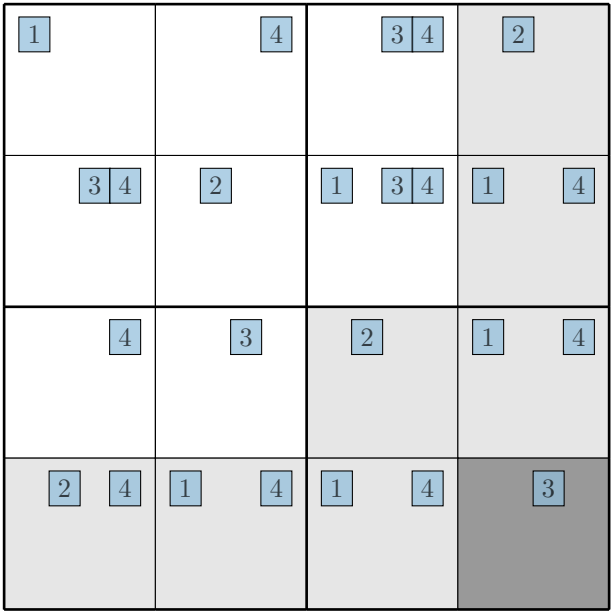


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

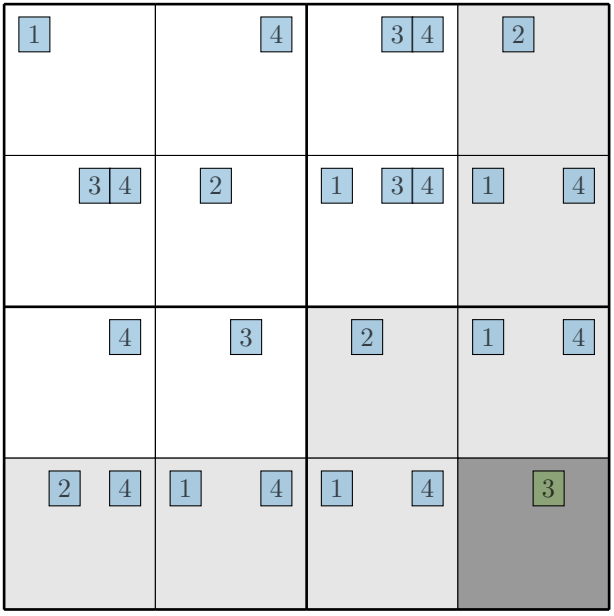


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

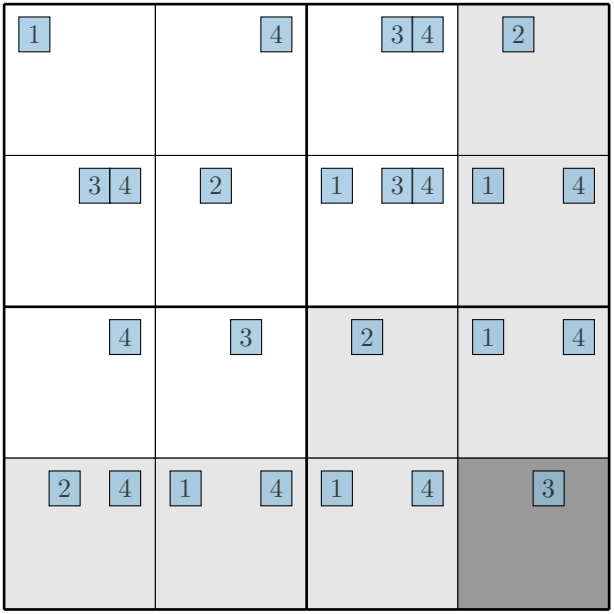


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

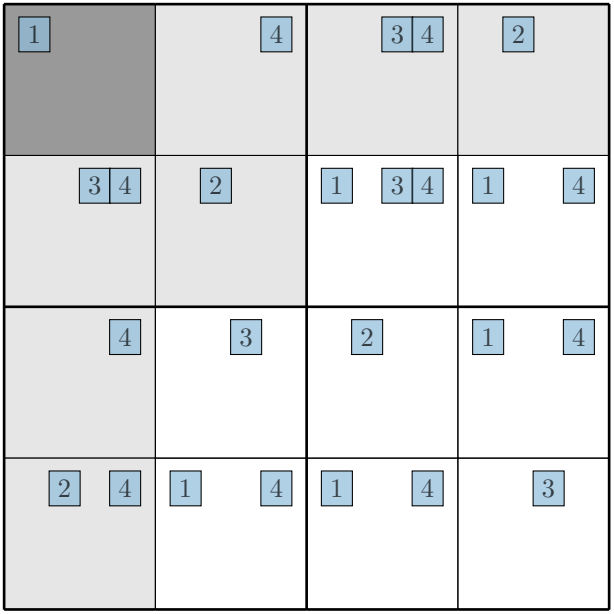


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

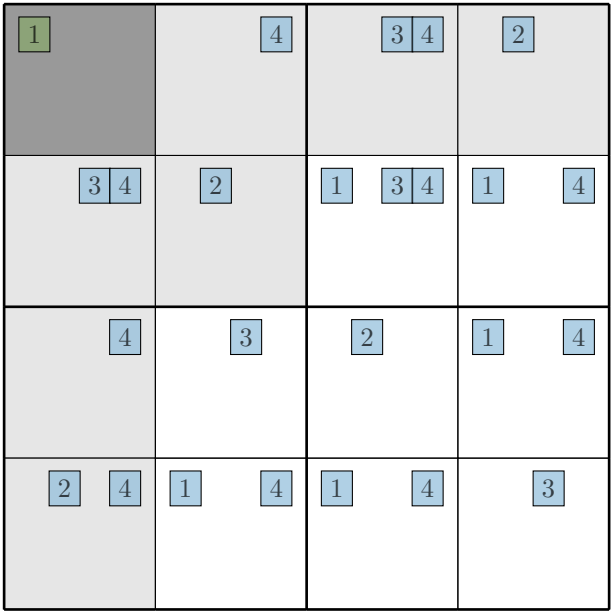


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

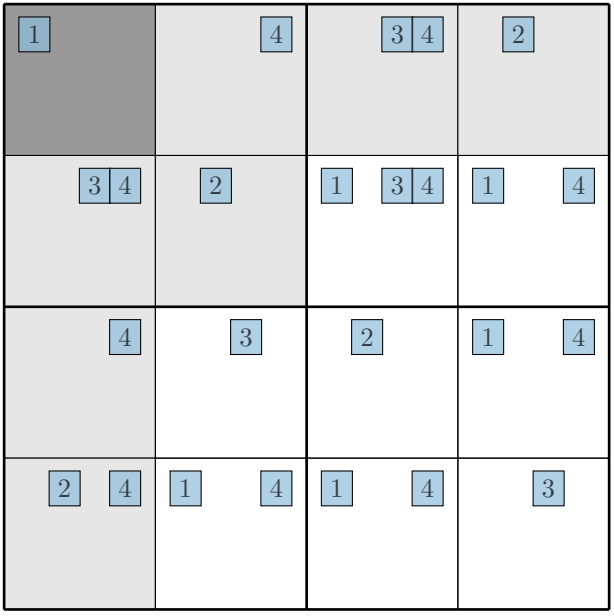


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

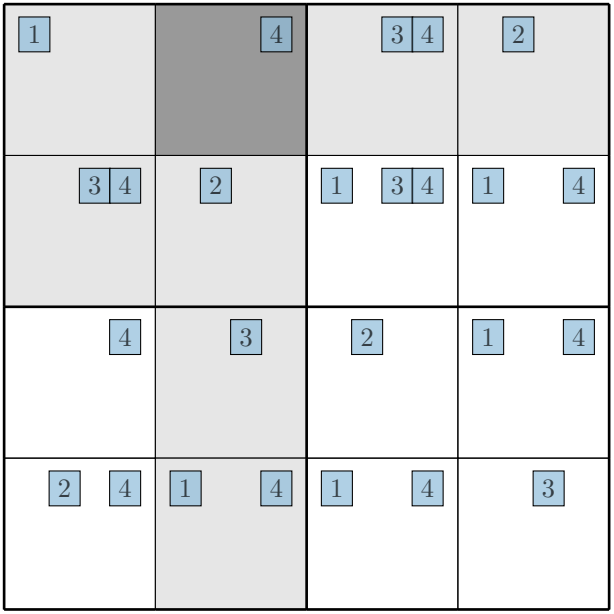


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

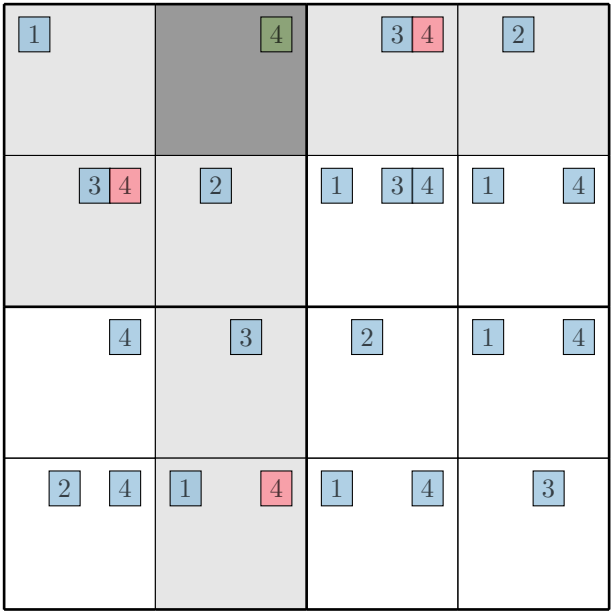


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

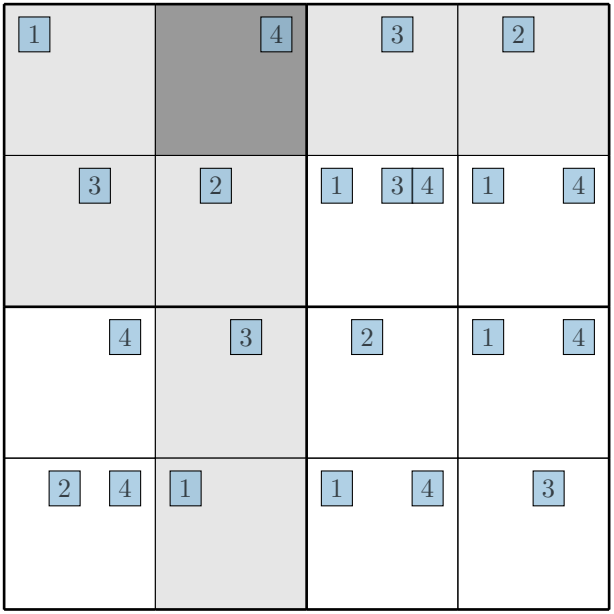


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

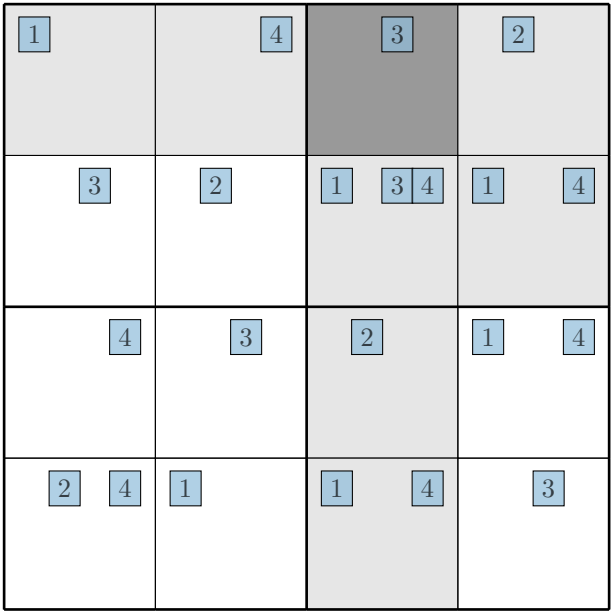


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

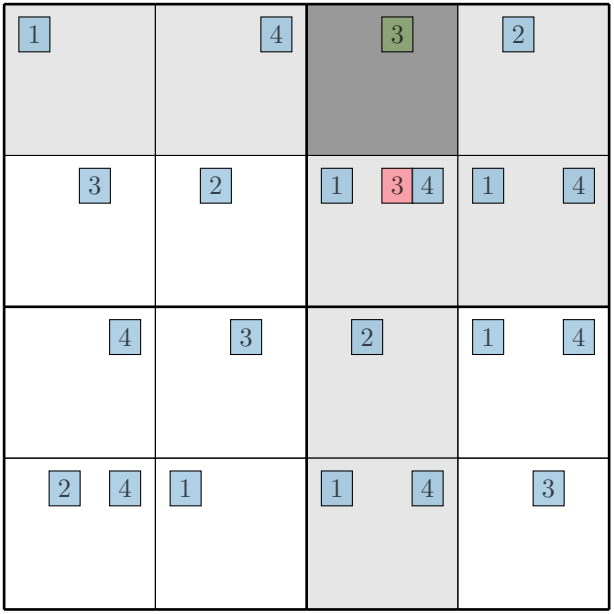


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

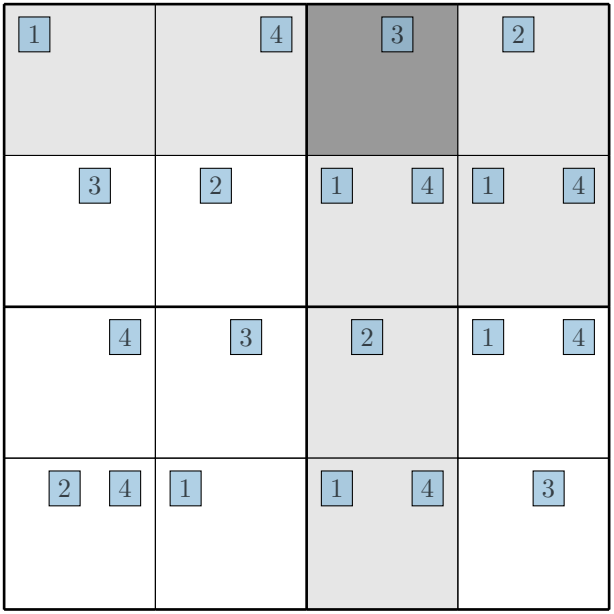


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

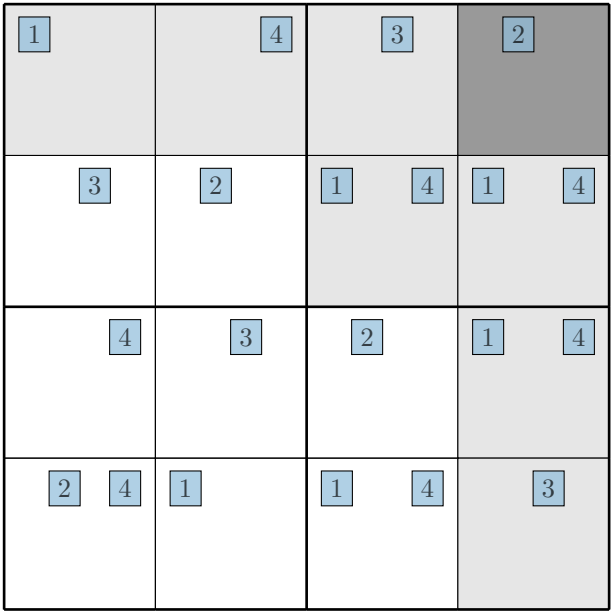


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

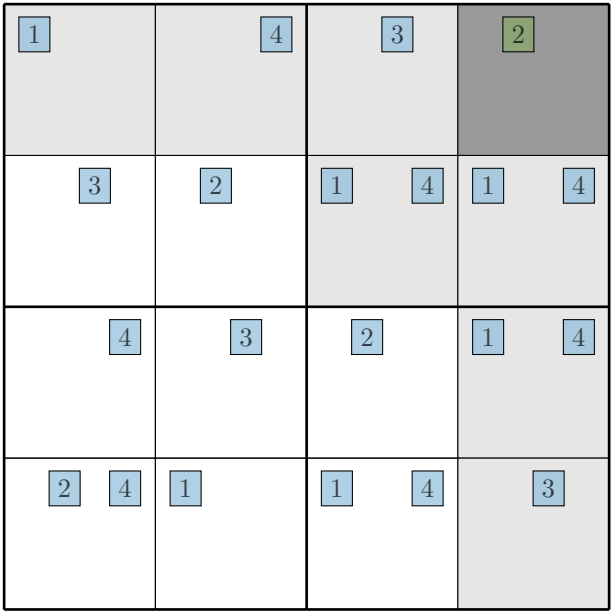


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

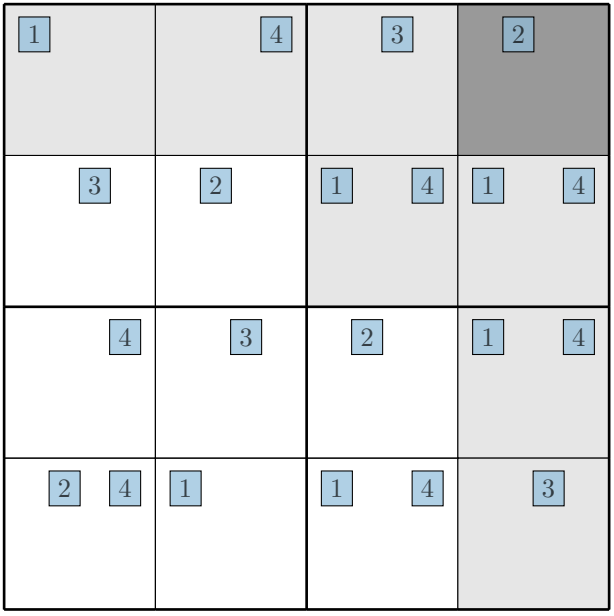


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

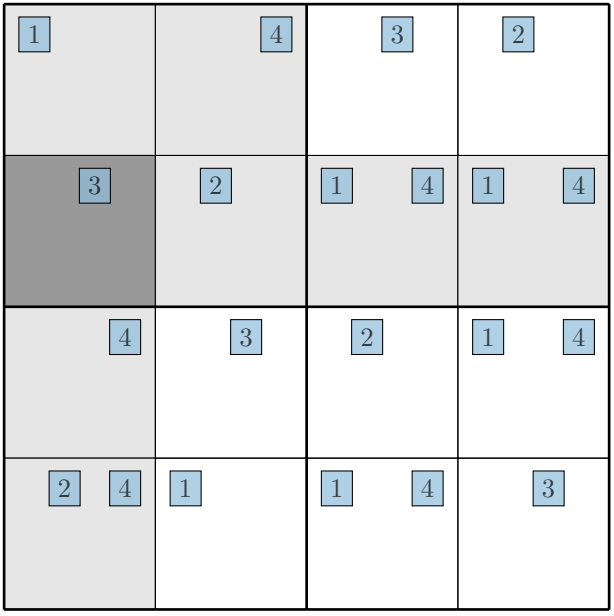


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

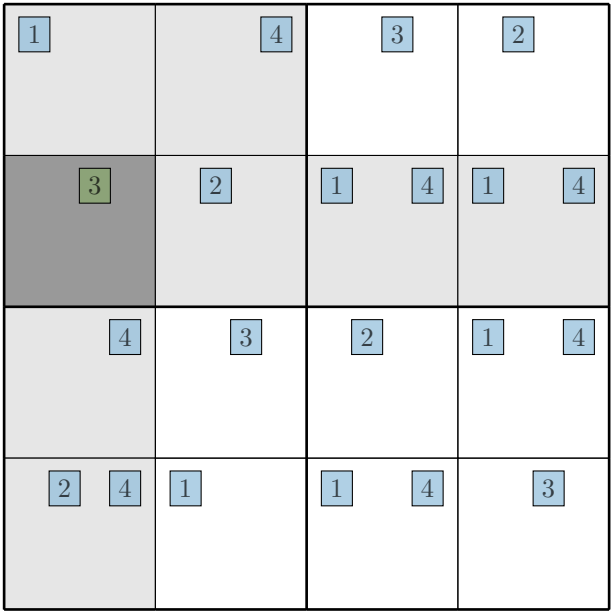


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

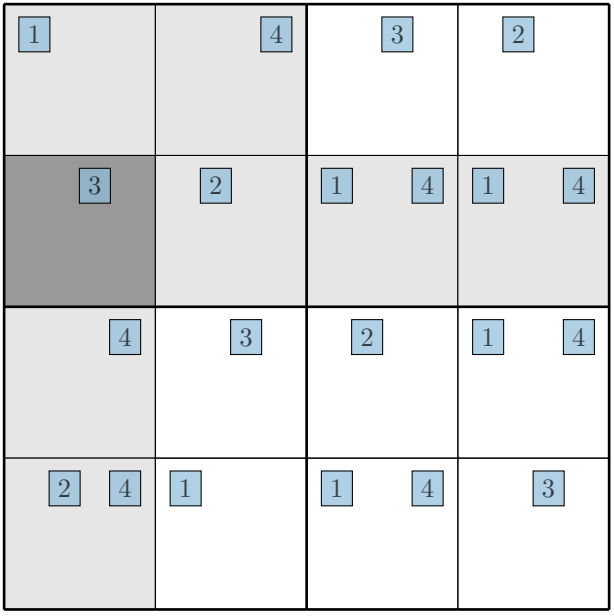


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

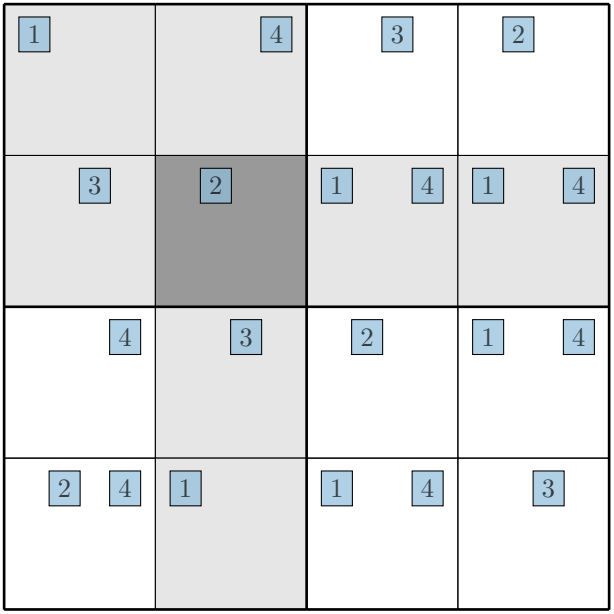


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

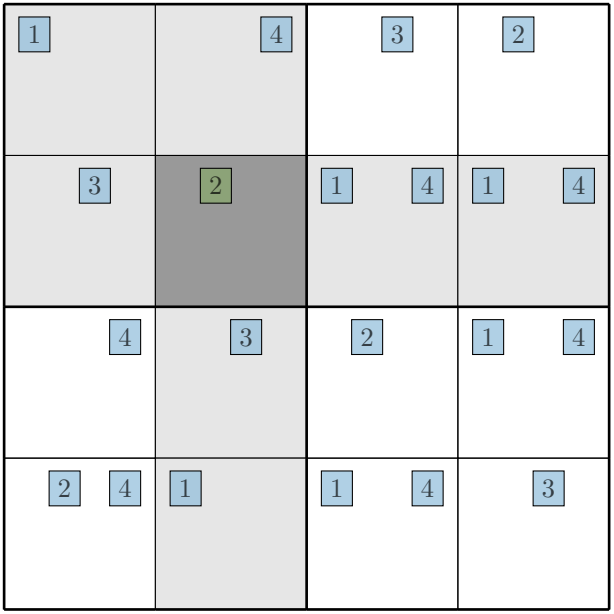


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

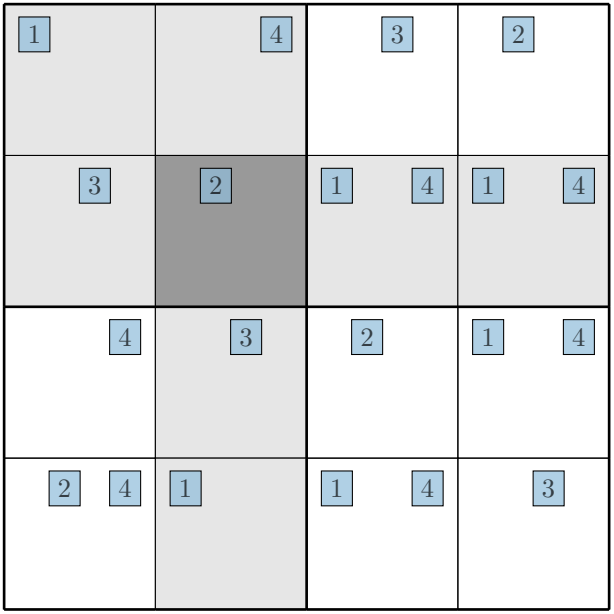


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

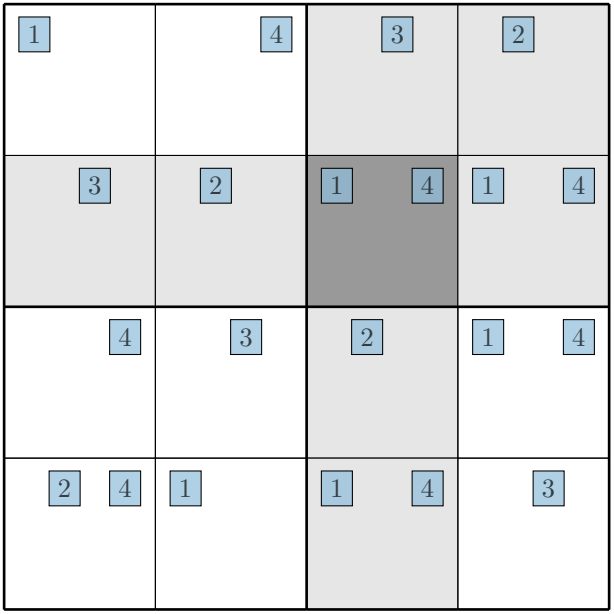


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

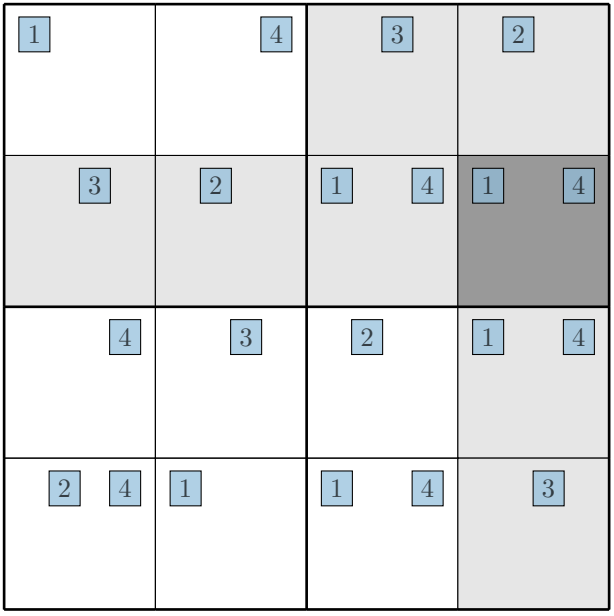


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

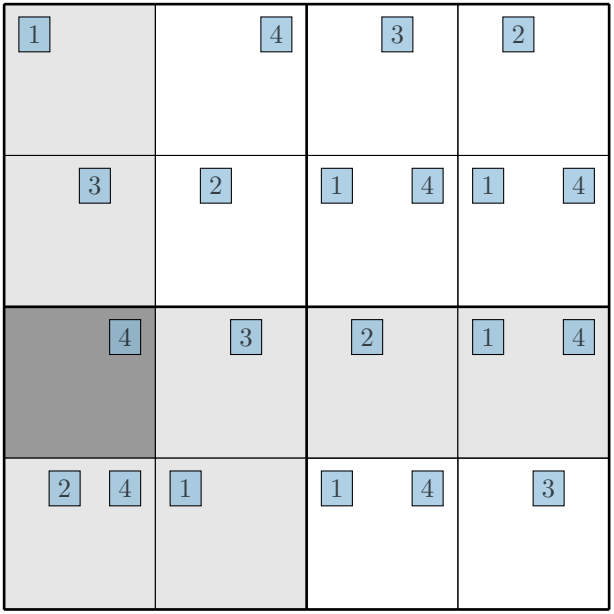


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

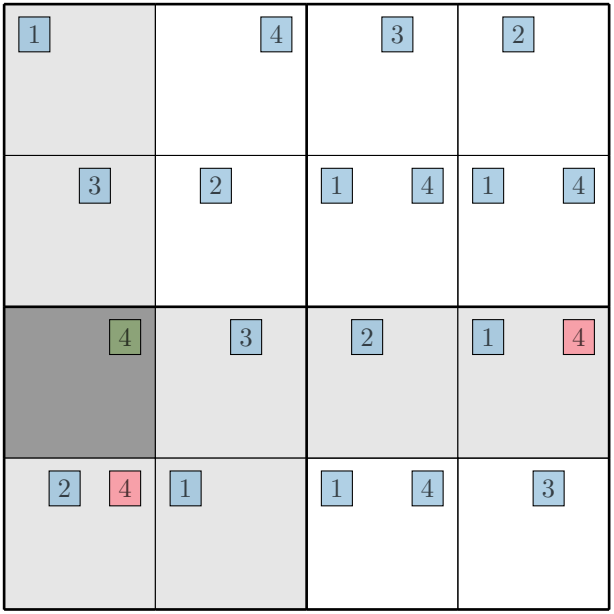


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

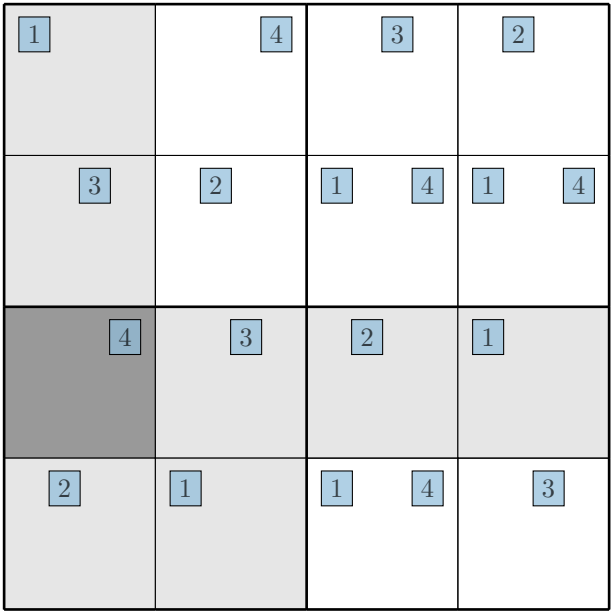


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

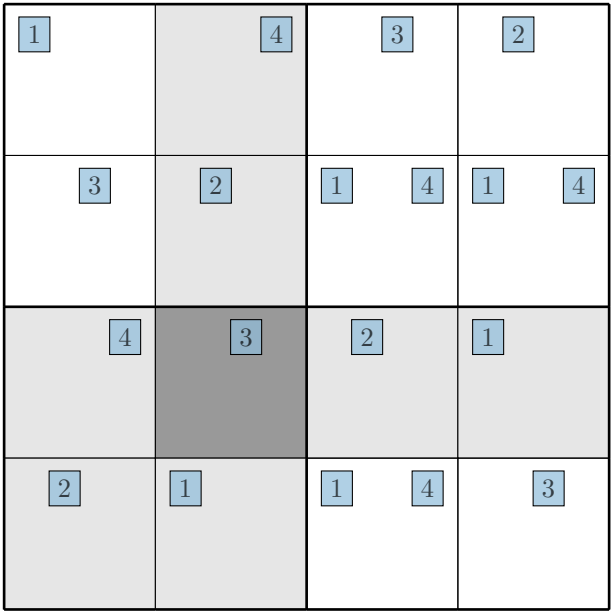


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

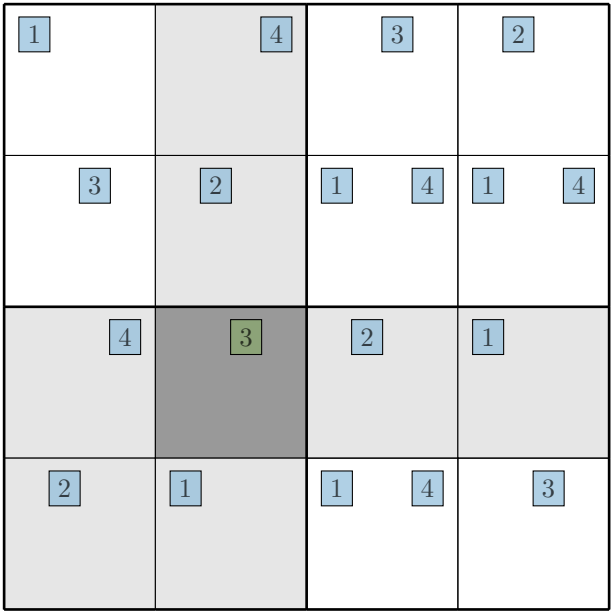


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

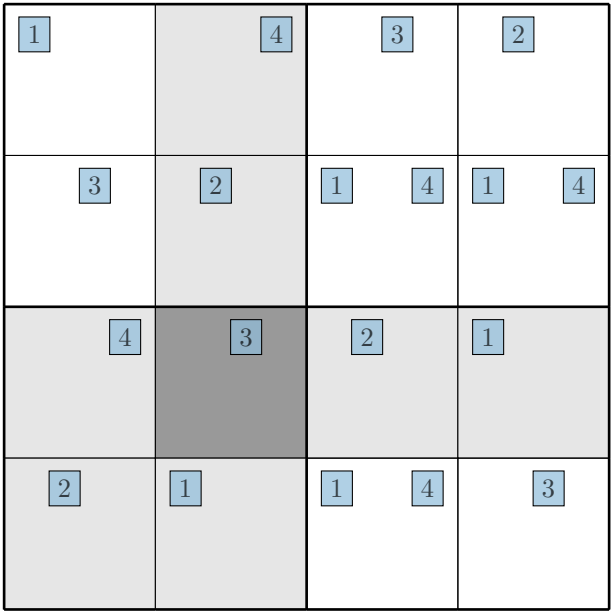


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

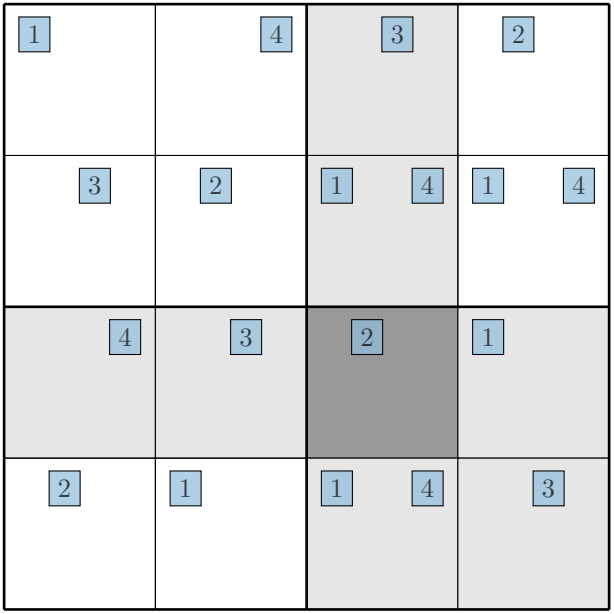


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

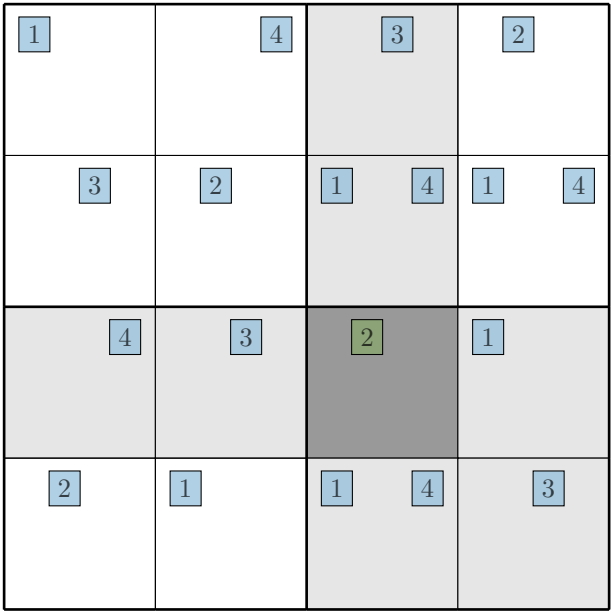


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

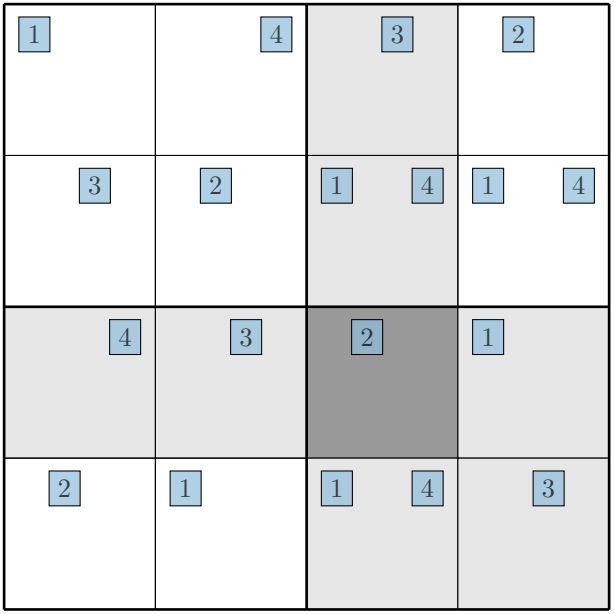


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

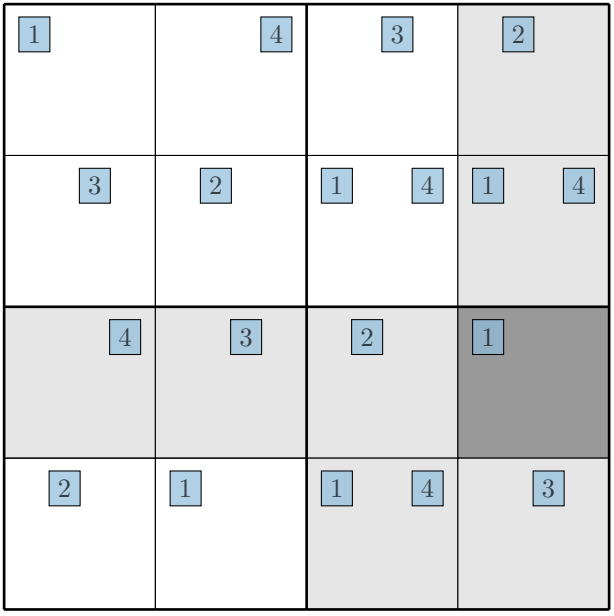


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

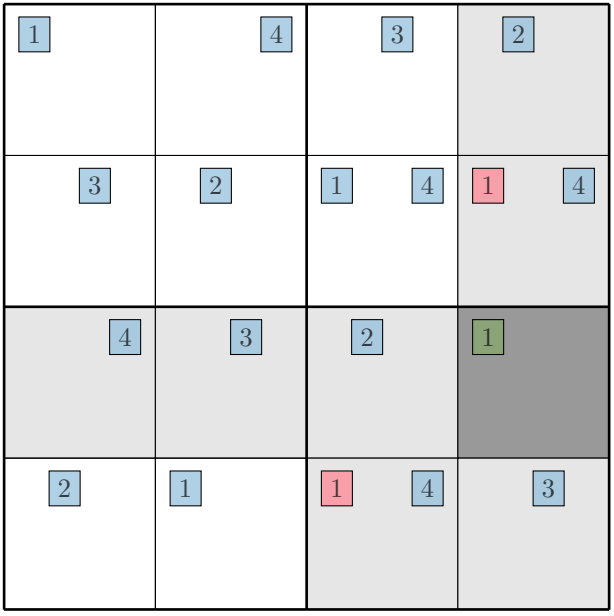


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

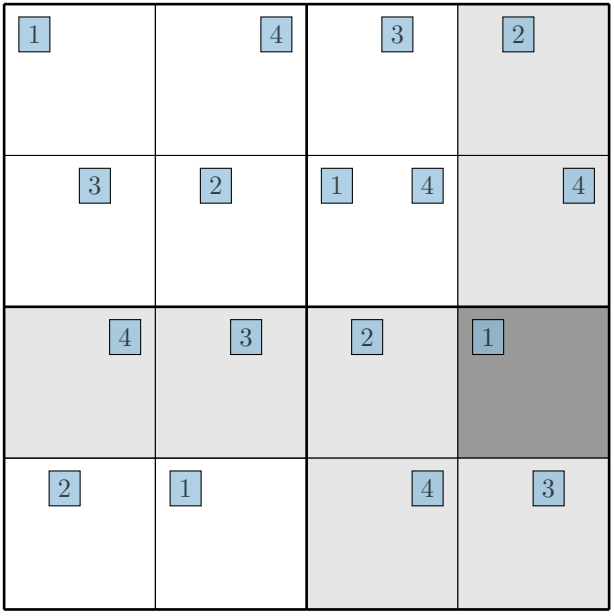


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

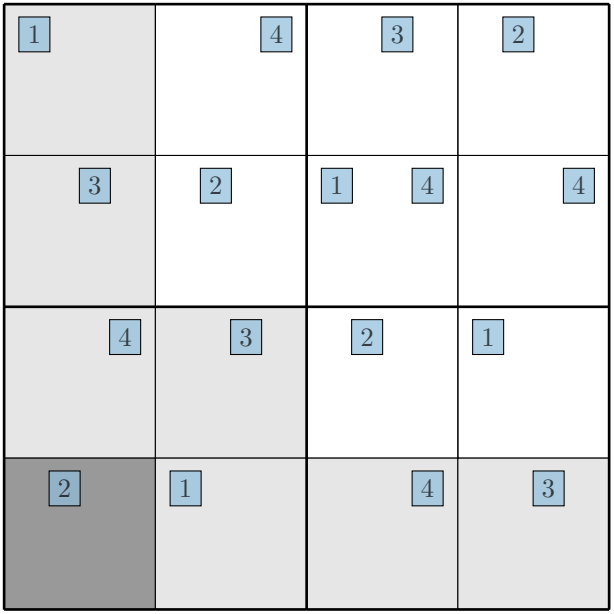


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

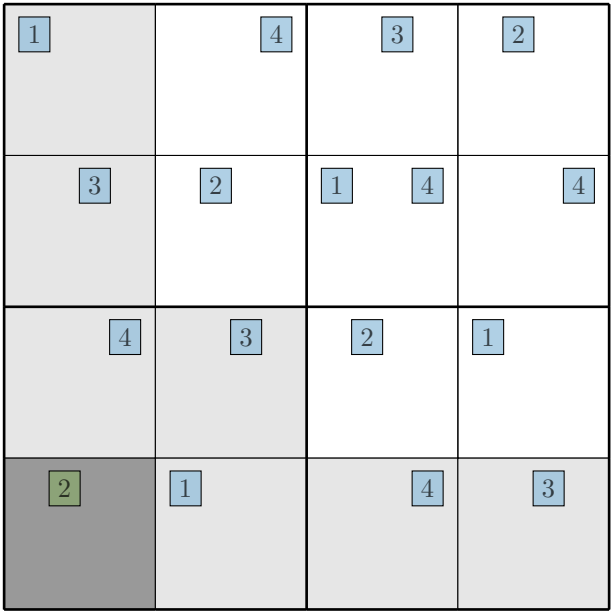


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

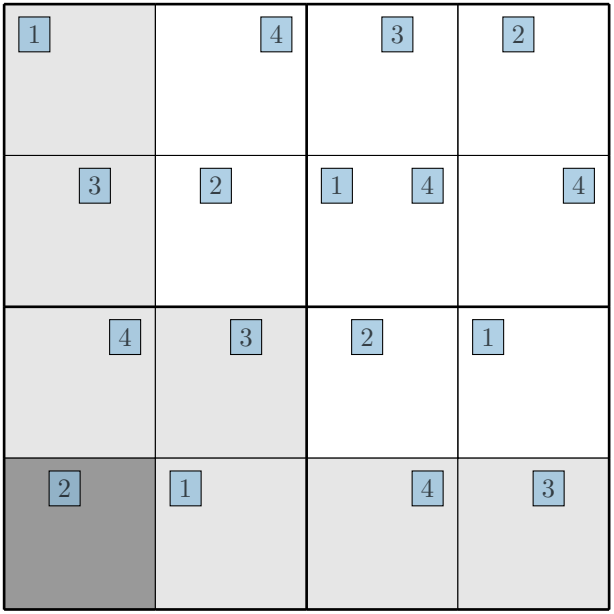


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

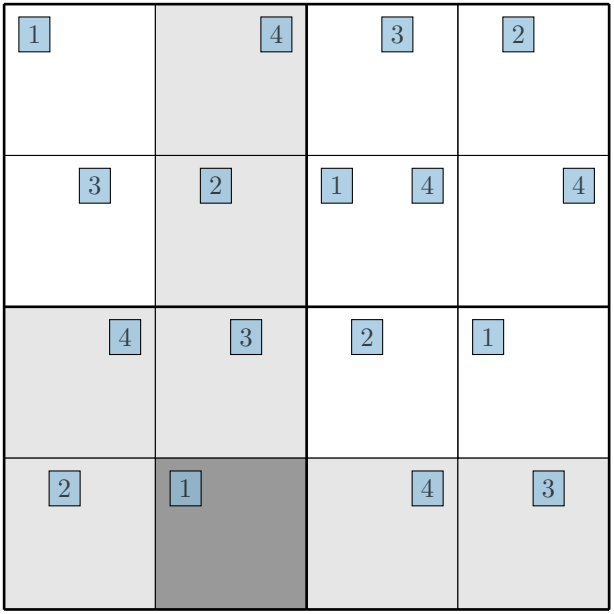


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

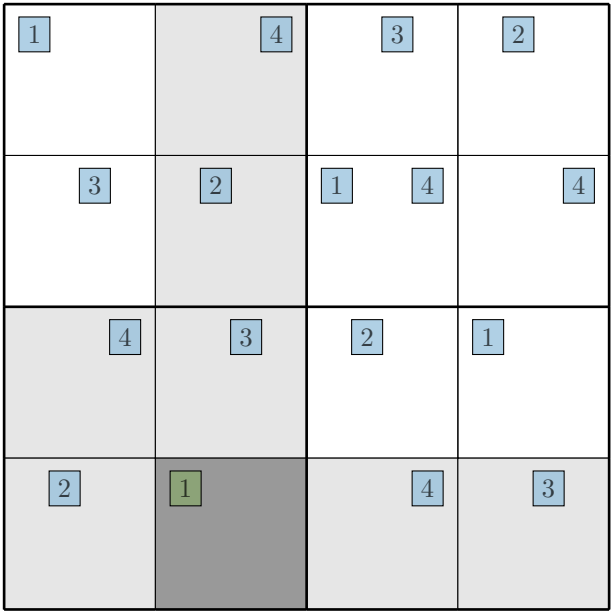


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

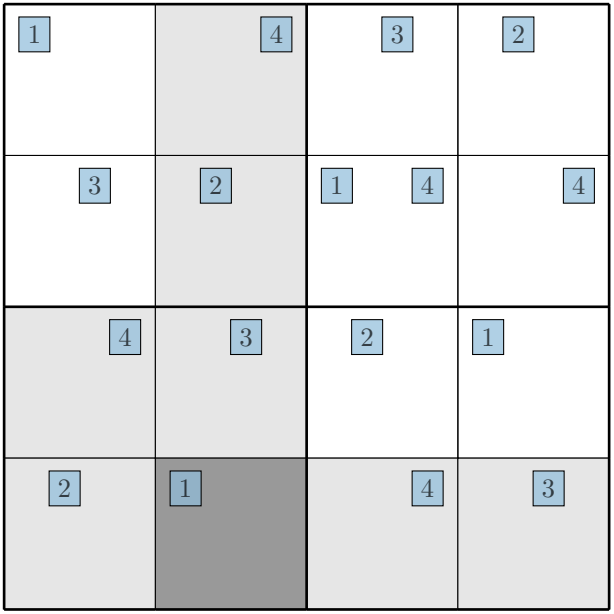


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

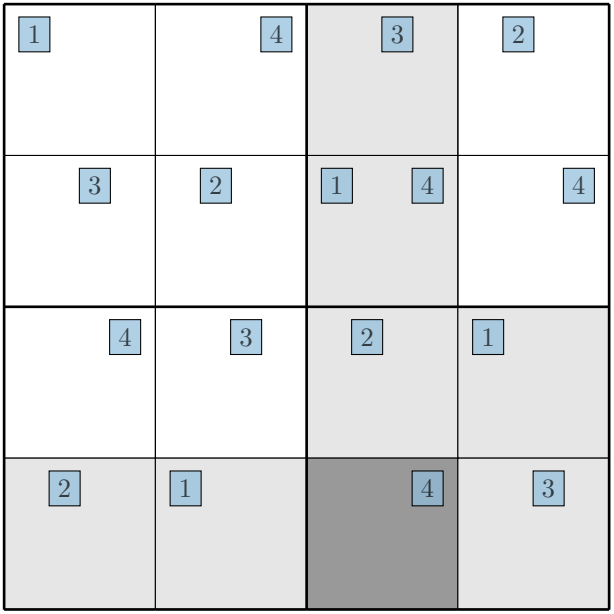


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

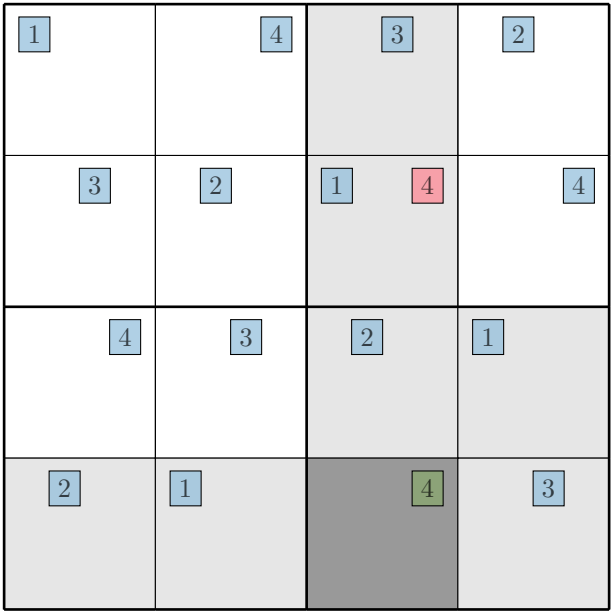


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

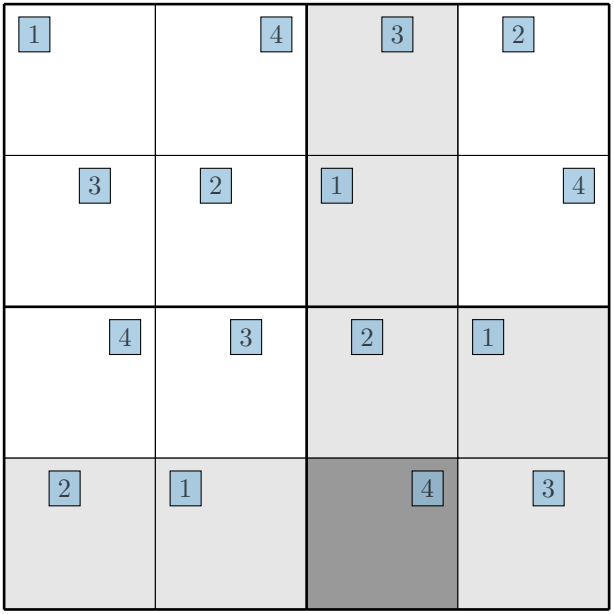


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

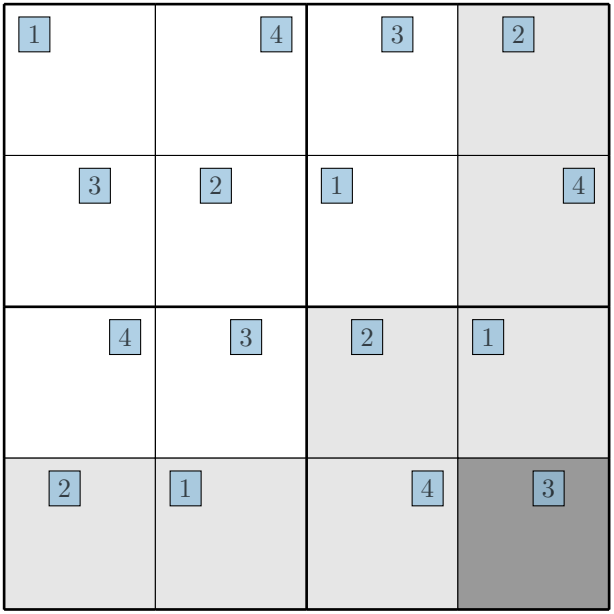


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

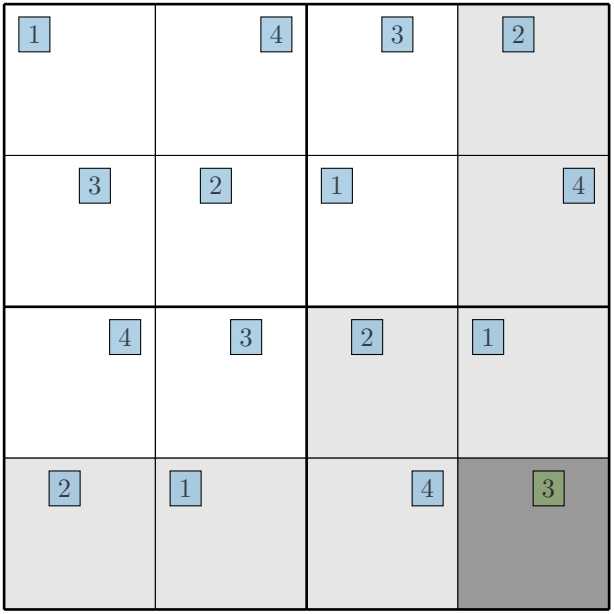


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

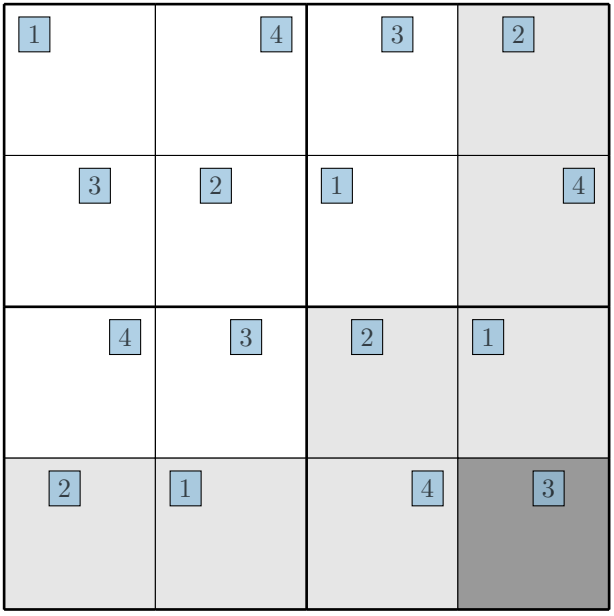


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

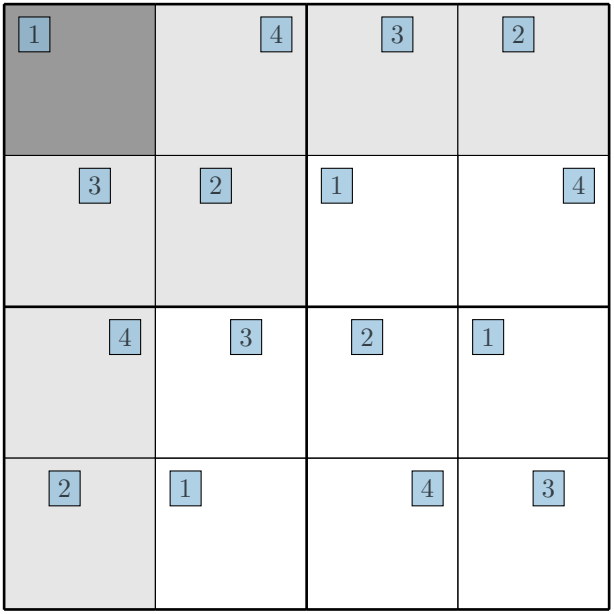


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

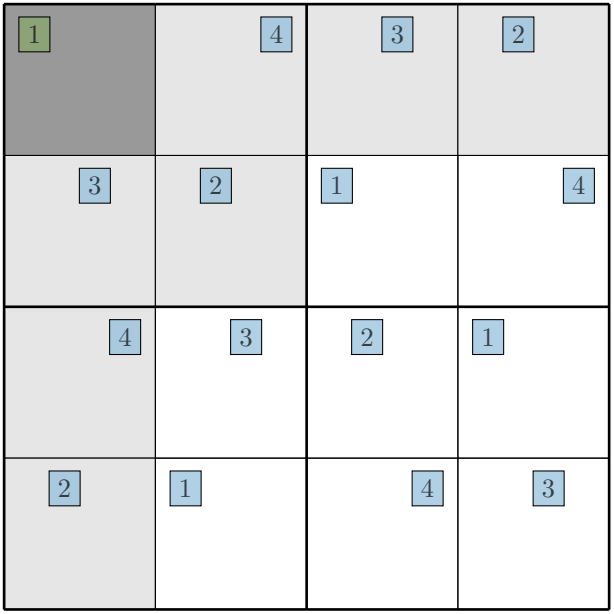


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

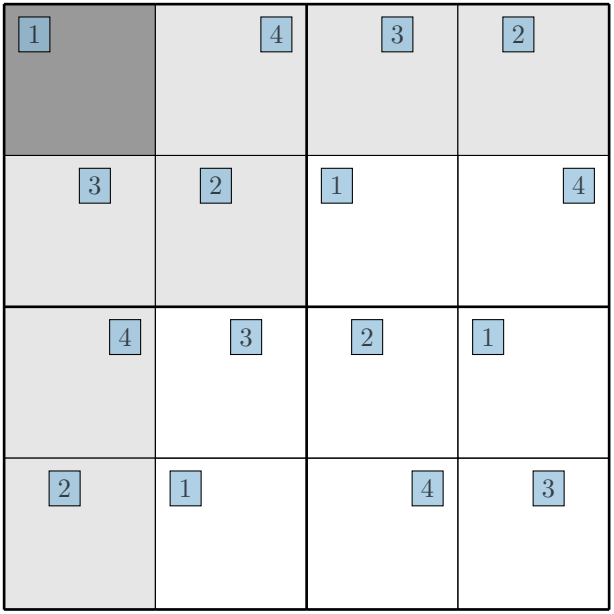


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

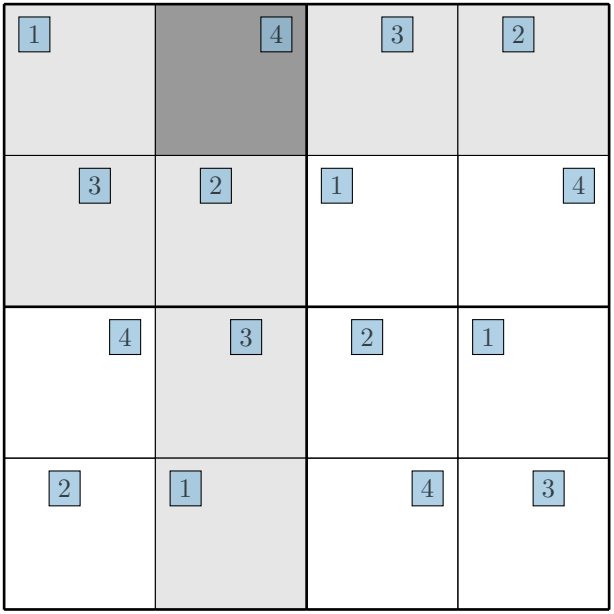


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

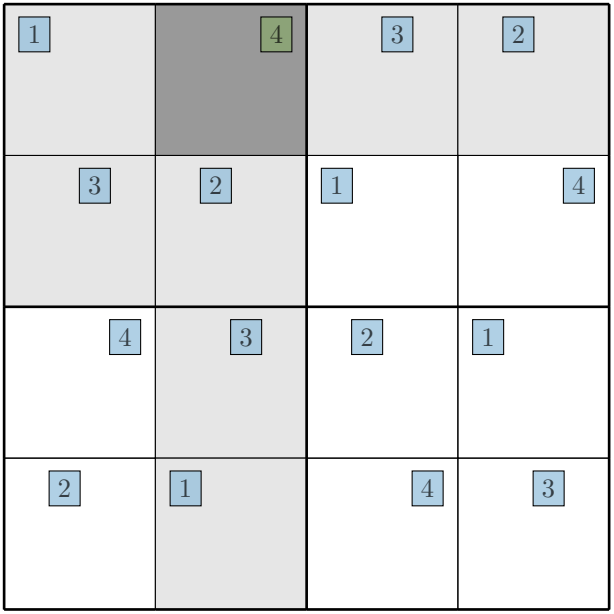


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

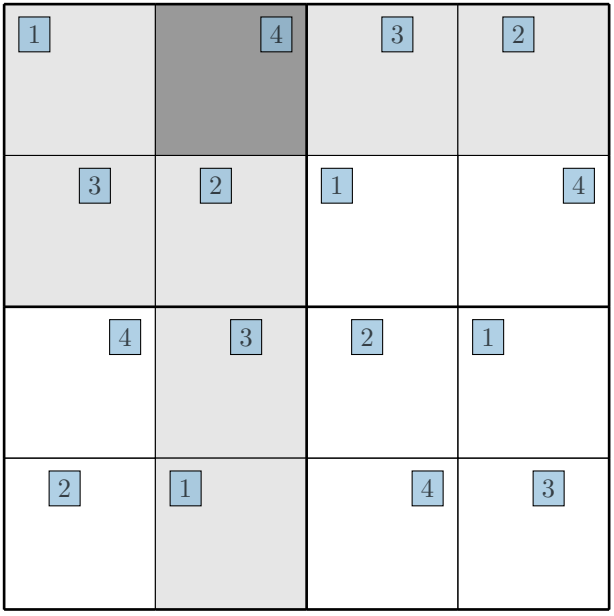


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

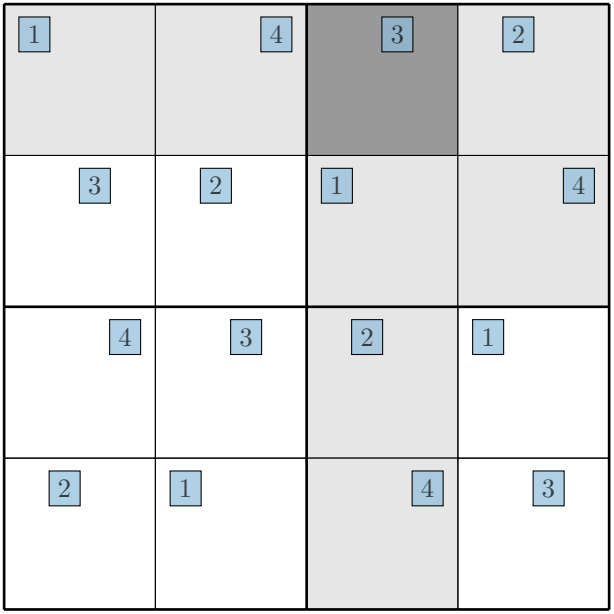


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

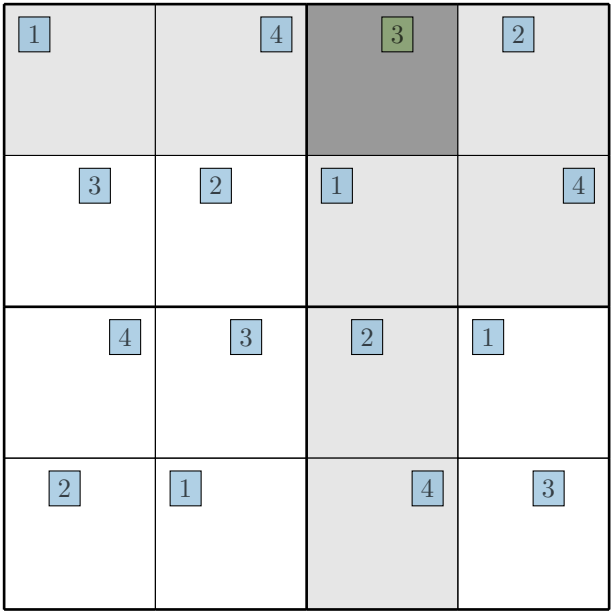


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

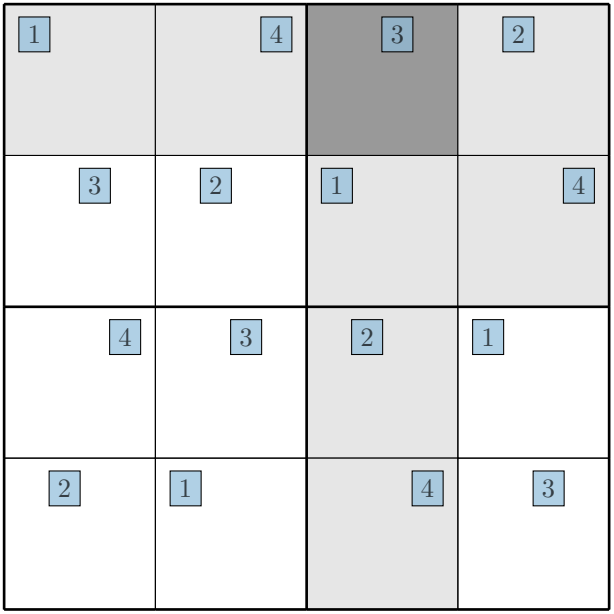


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

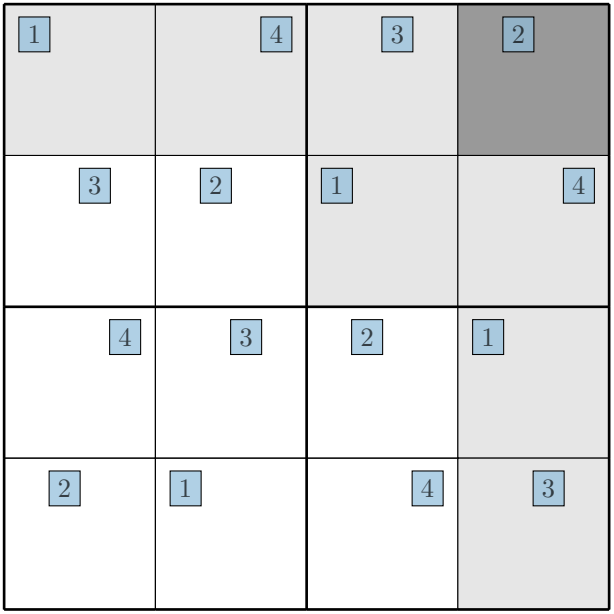


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

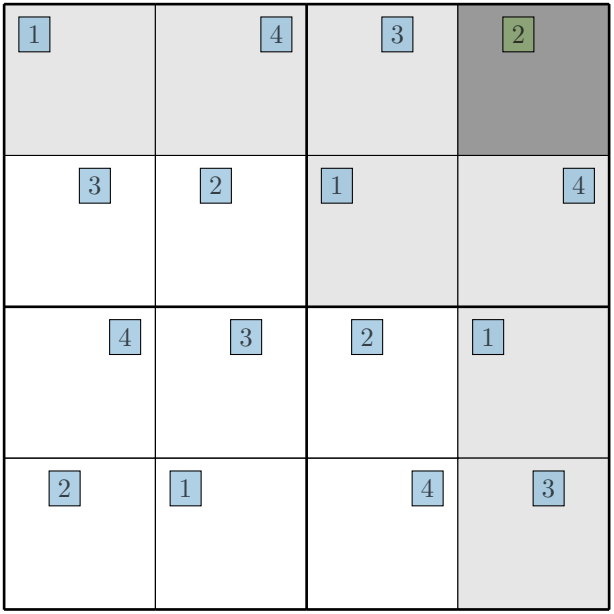


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

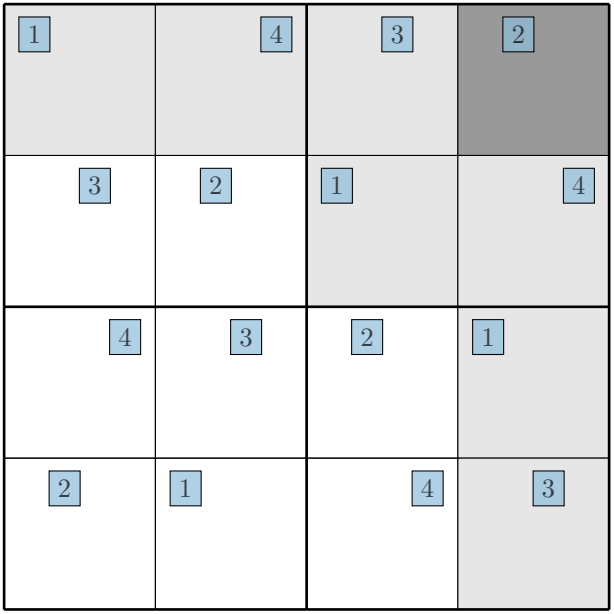


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

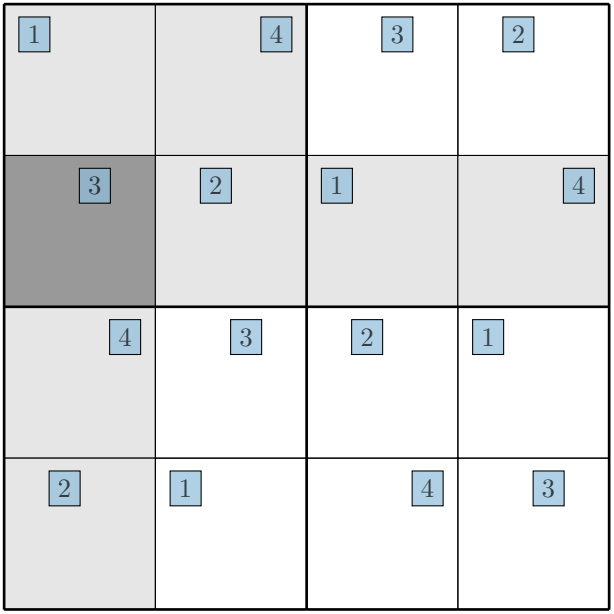


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

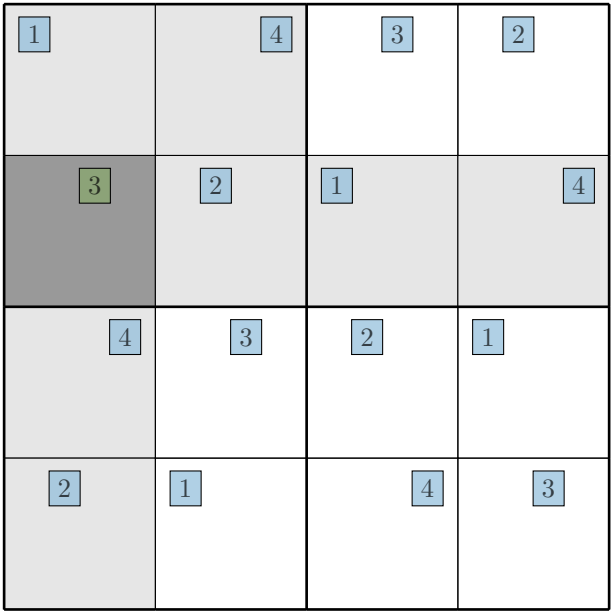


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

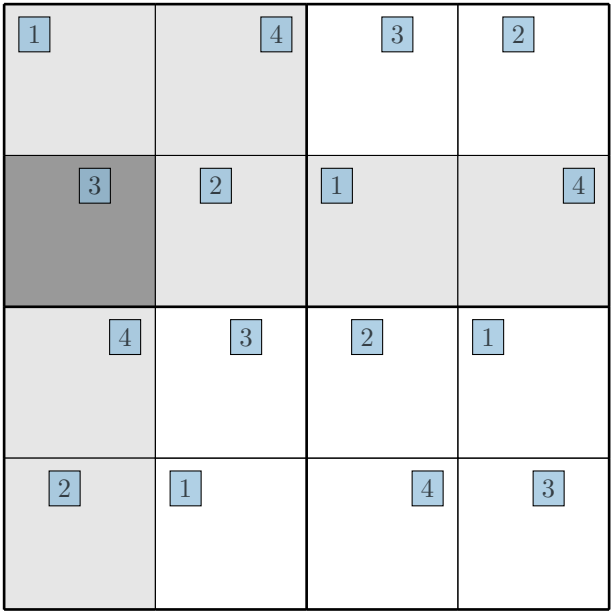


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

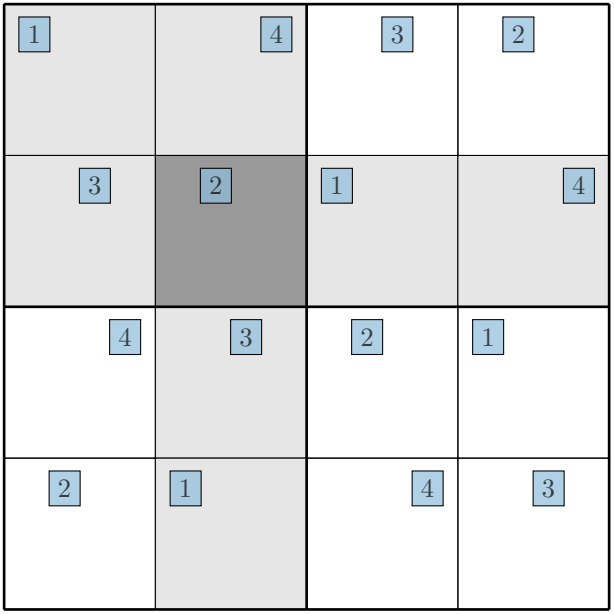


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

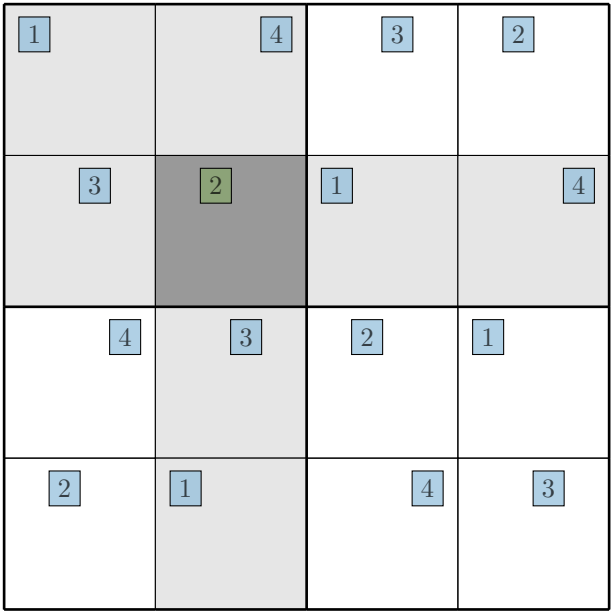


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

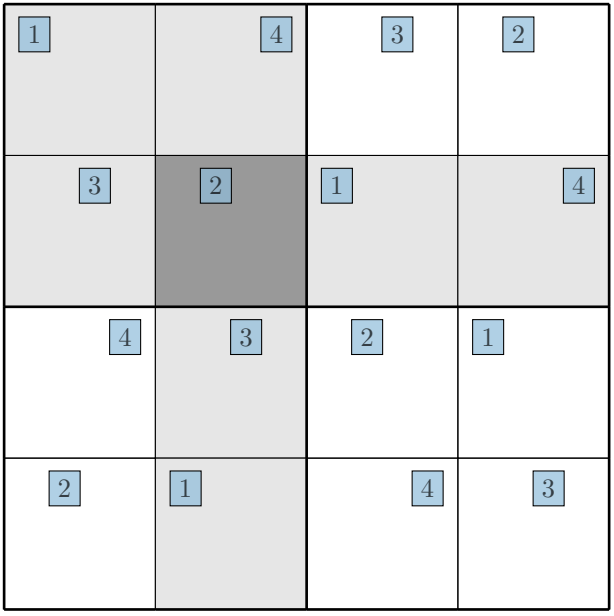


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

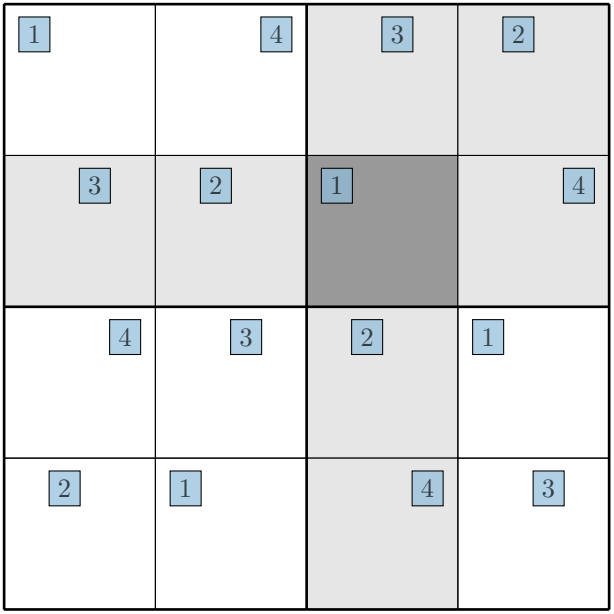


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

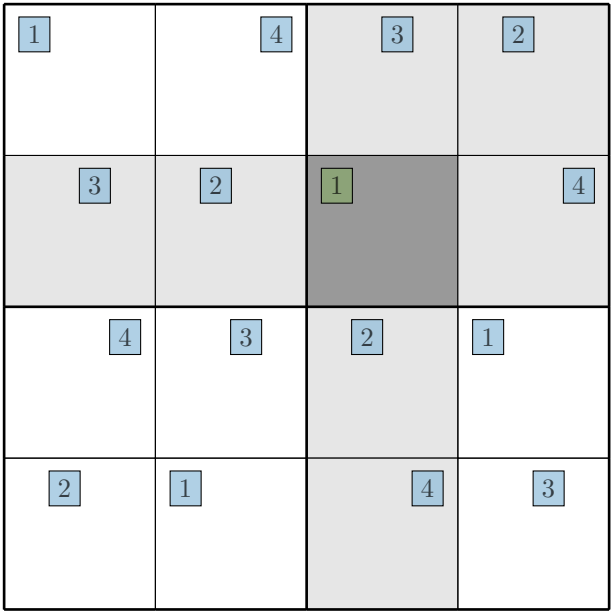


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

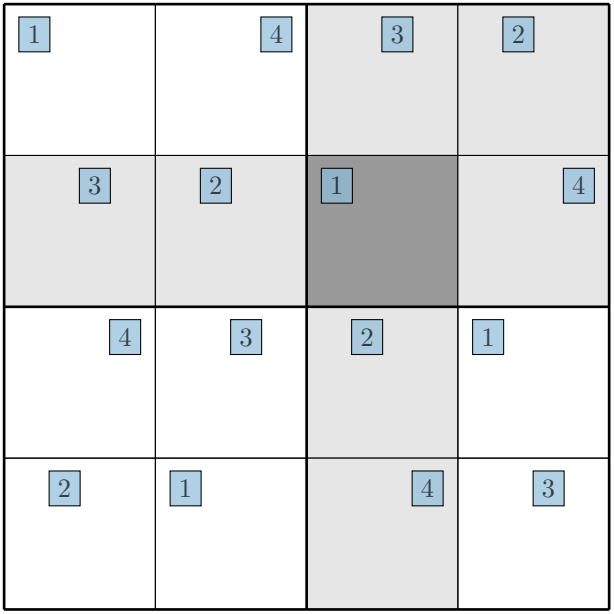


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

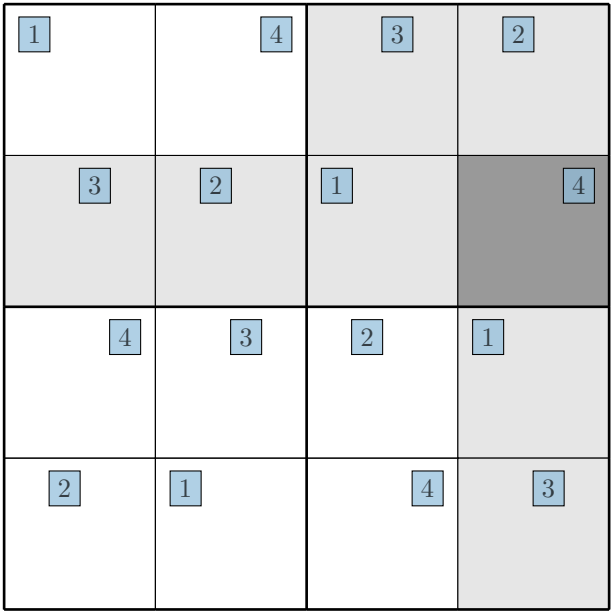


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

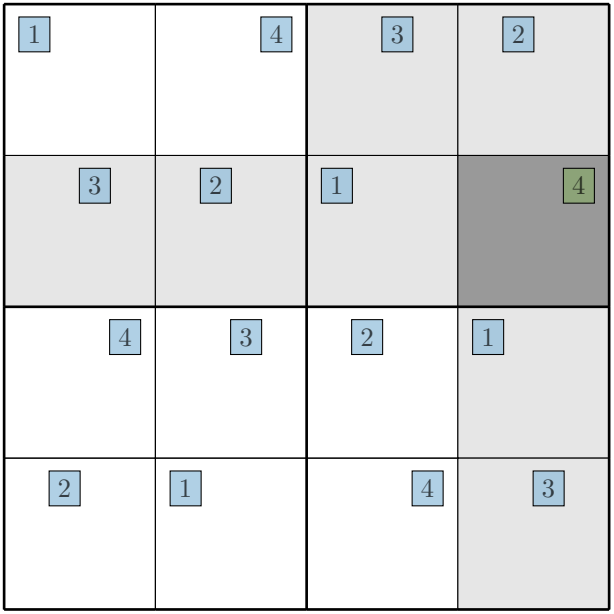


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

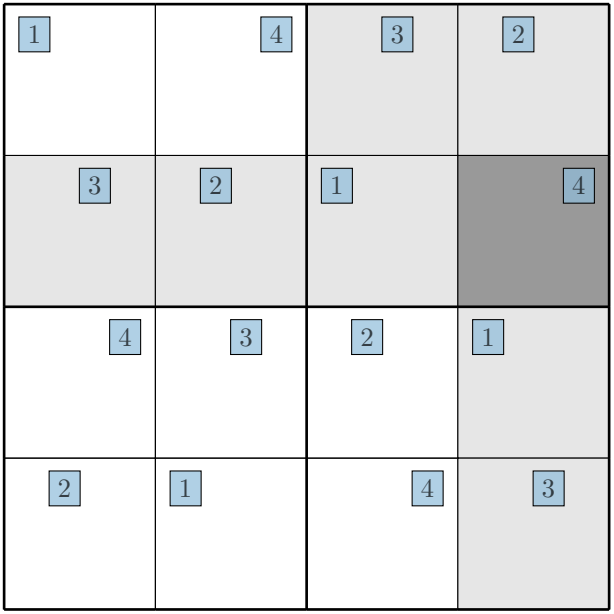


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

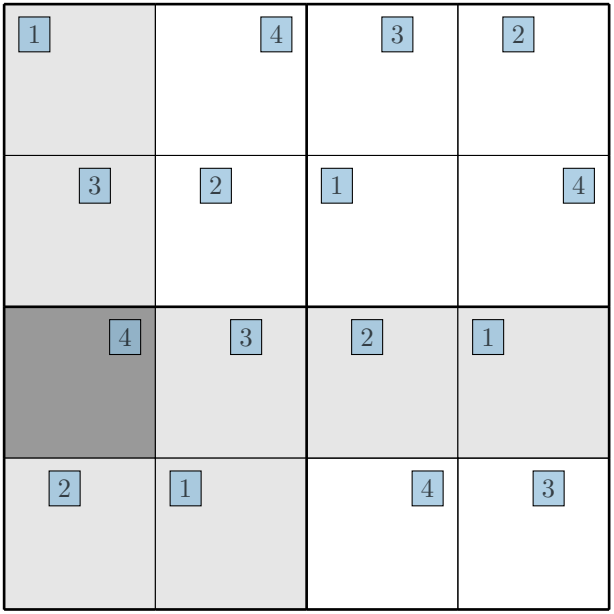


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

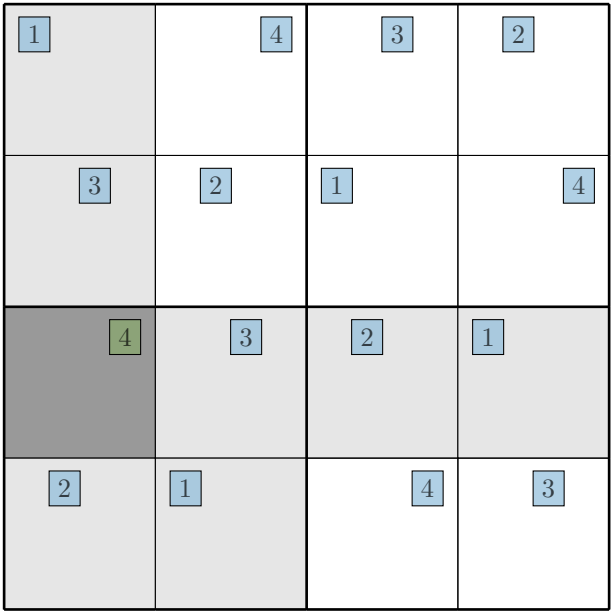


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

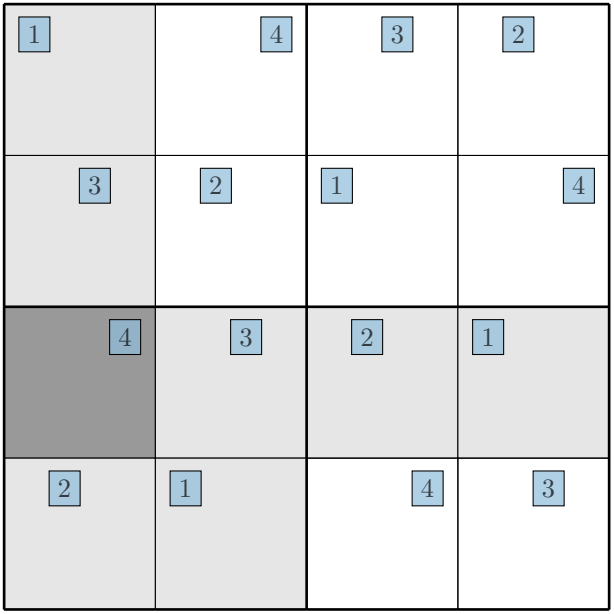


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

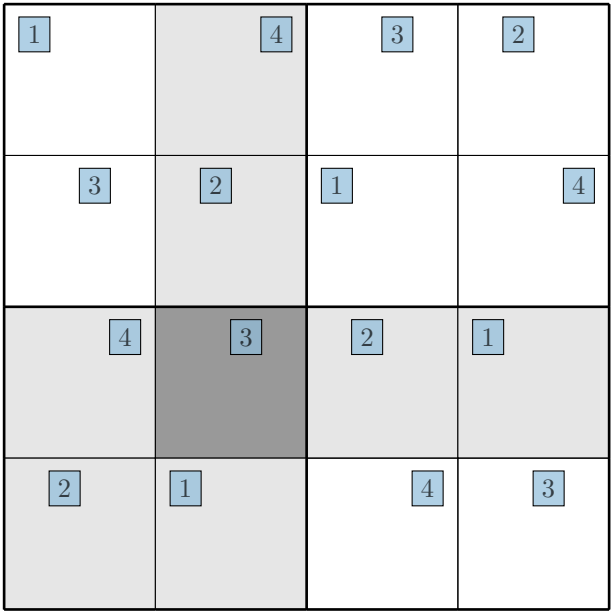


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

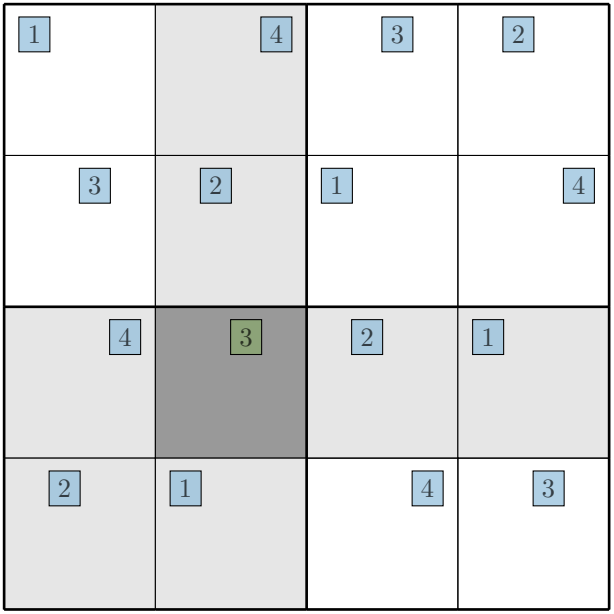


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

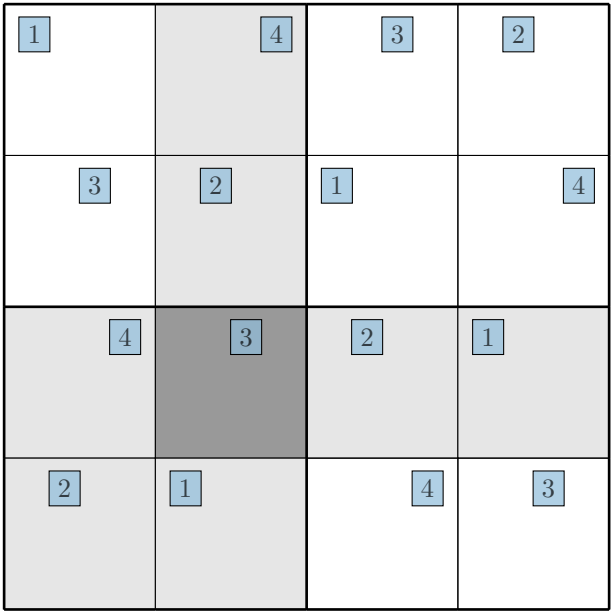


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

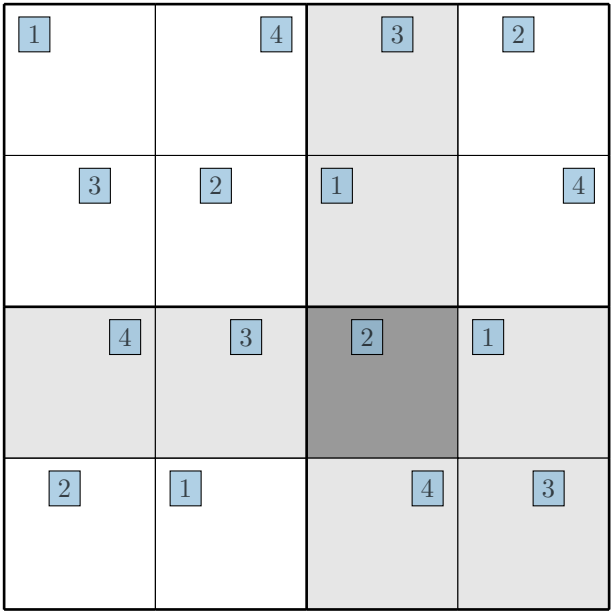


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

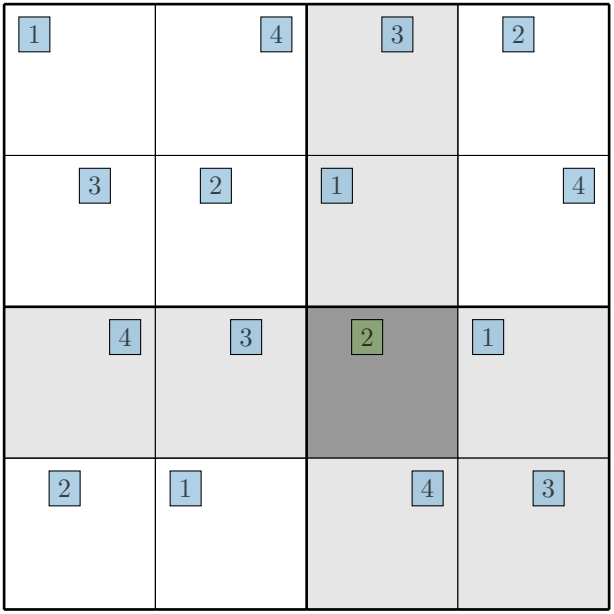


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

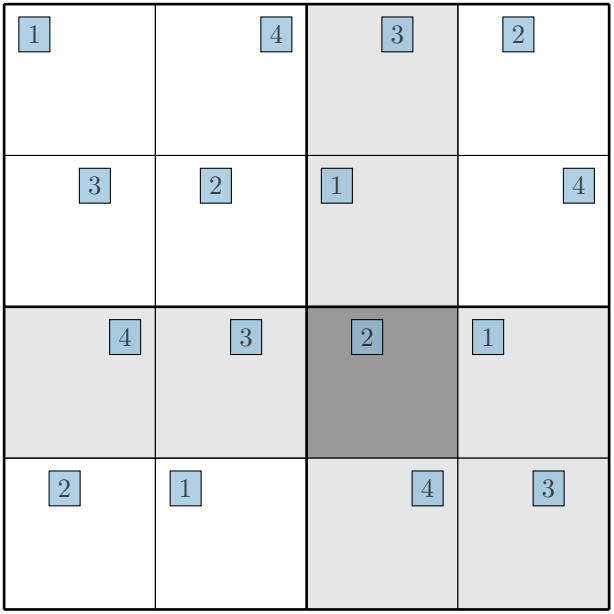


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

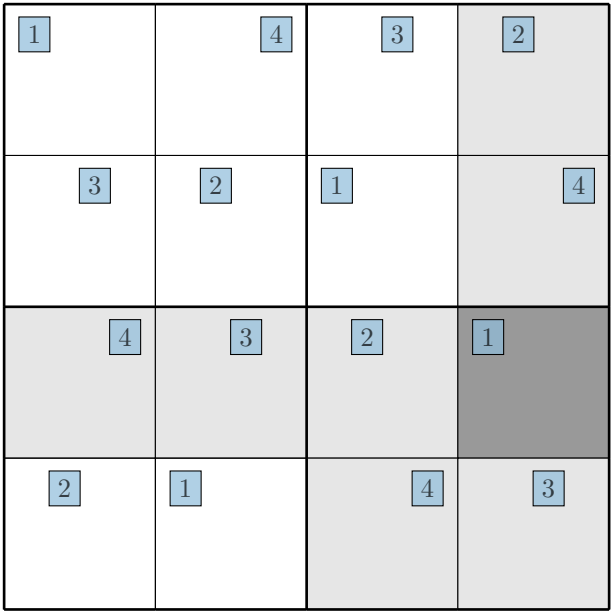


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

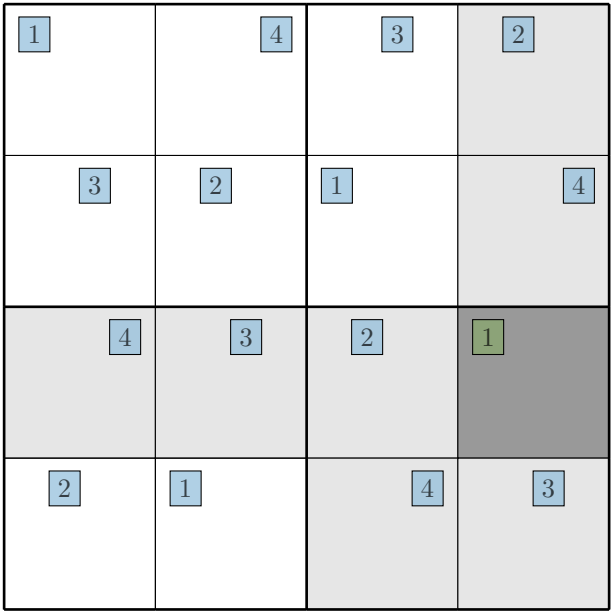


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

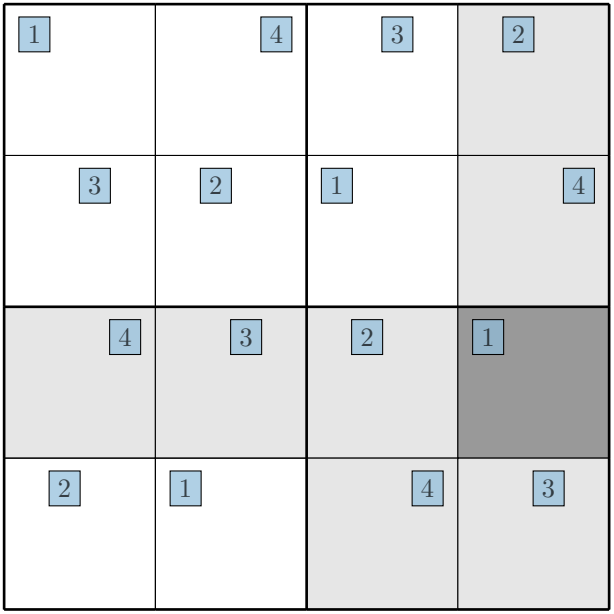


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

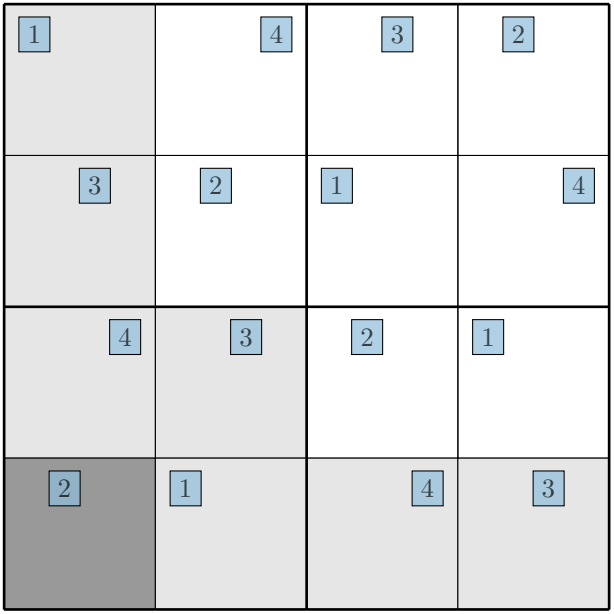


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

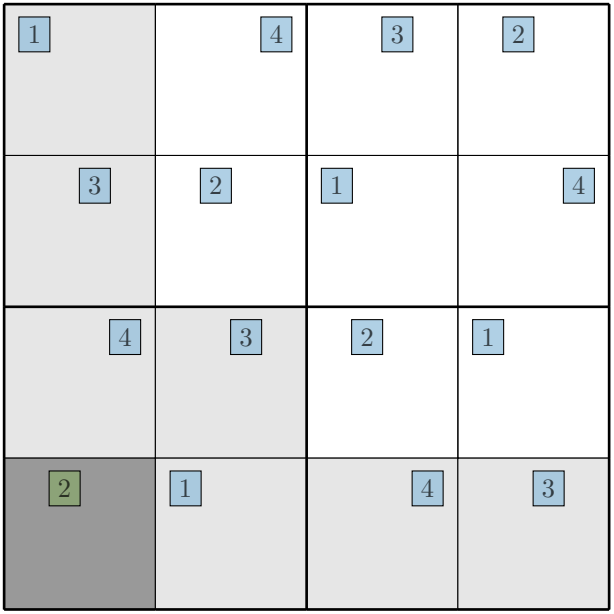


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if |`domains[spot]`| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains[spot]`| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

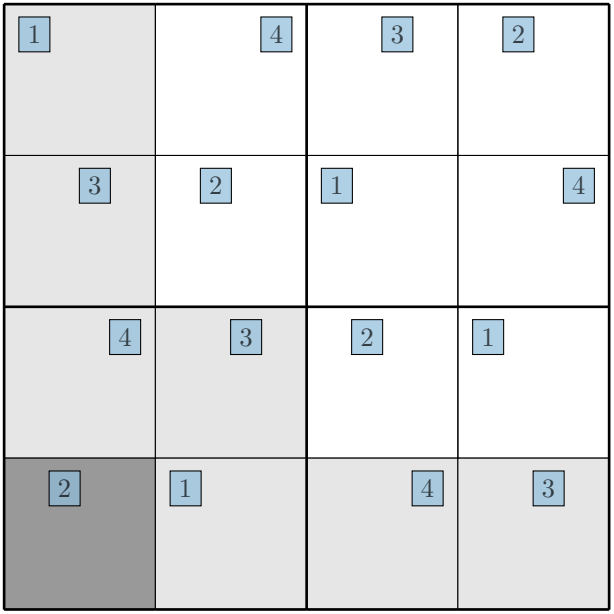


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

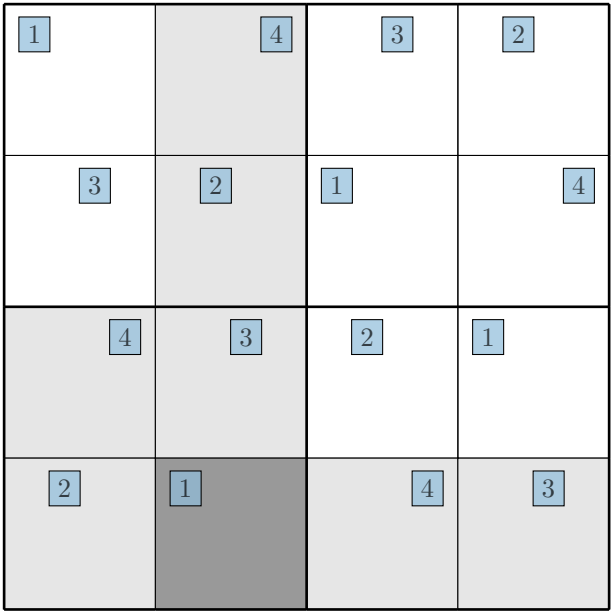


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

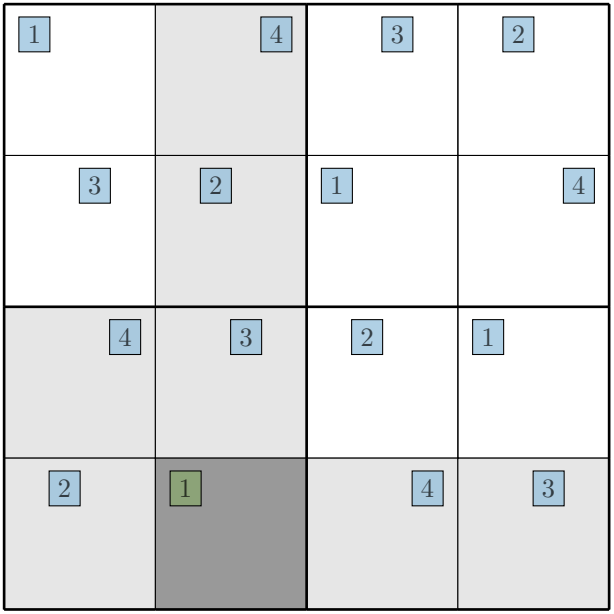


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

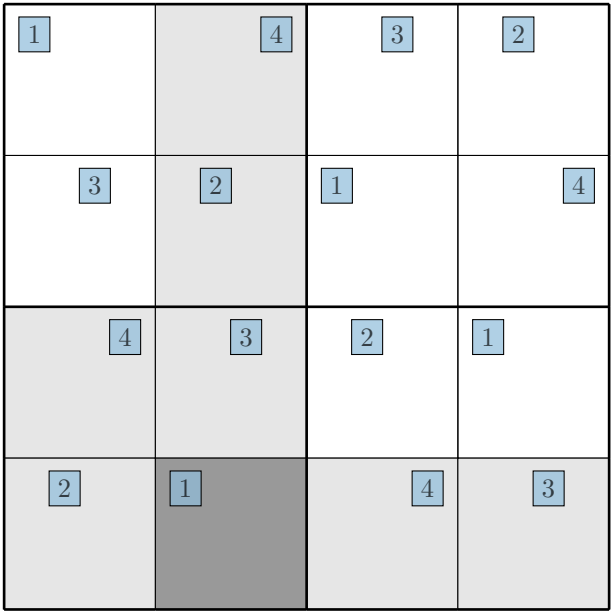


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

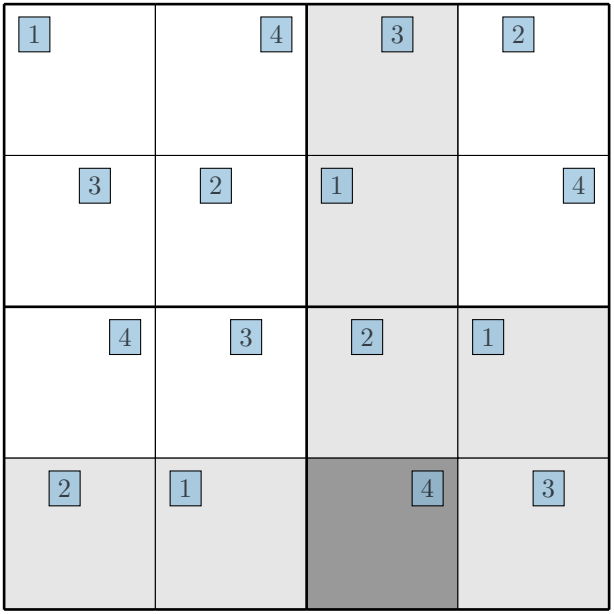


`solve(problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains(problem, domains)`
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate(domains)` reports no conflict:
    - if $|$`domains[spot]`$| = 1$ for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy(domains)`
    - `spot, num` = `search(domains)`
    - add (`spot, num, orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack(stack)`

`propagate(domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if $|$`domains[spot]`$| = 1$, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

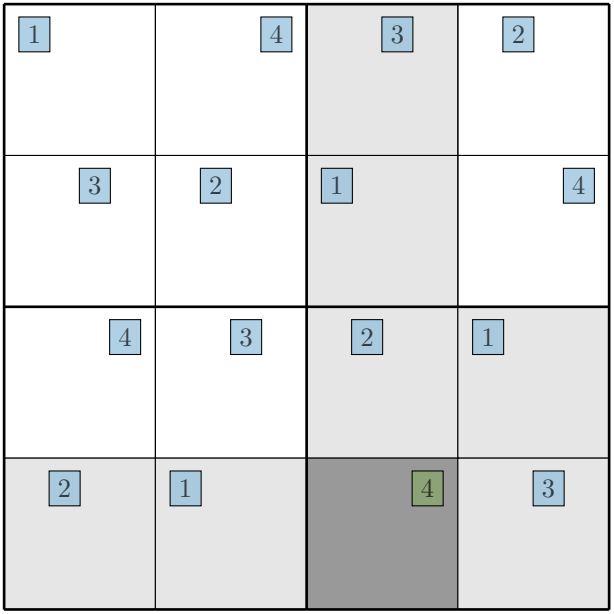


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

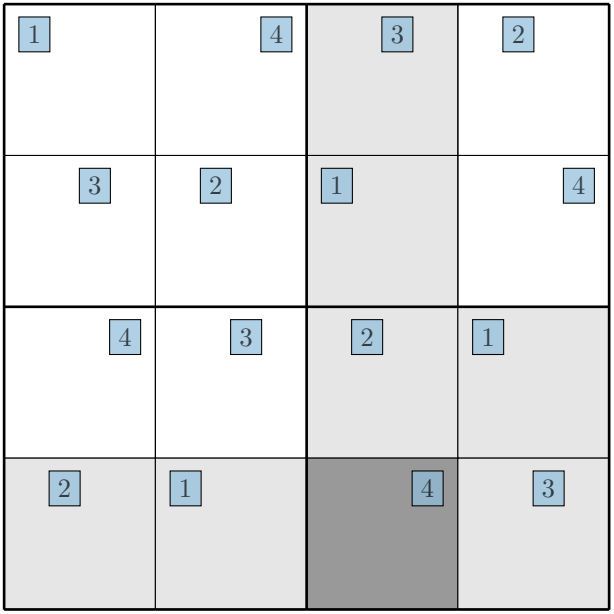


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

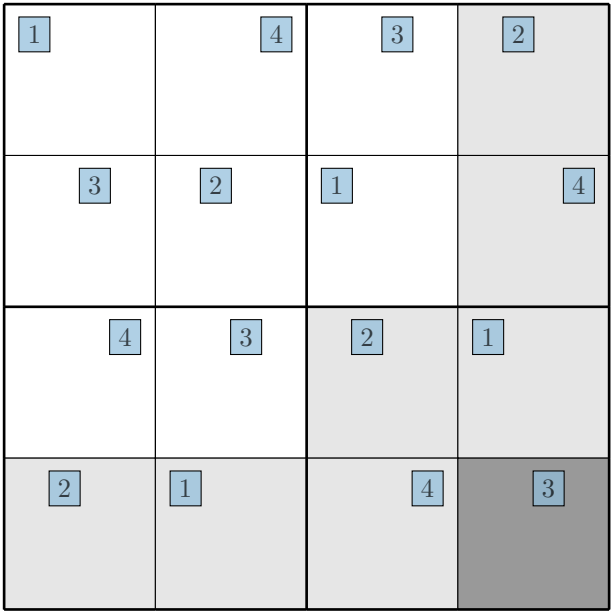


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

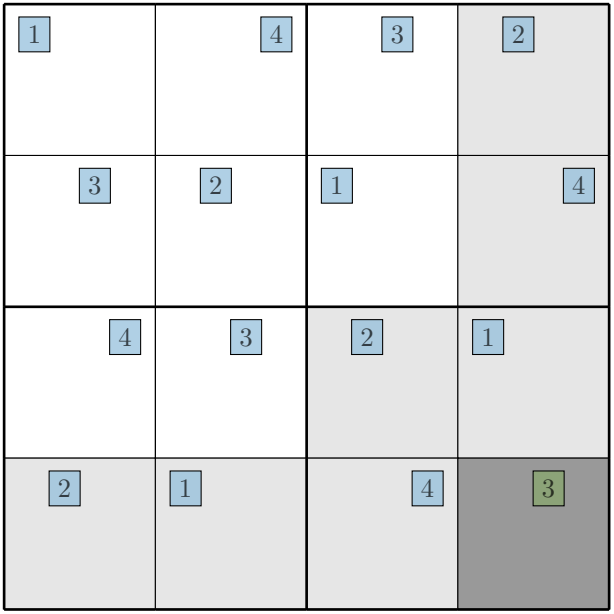


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict

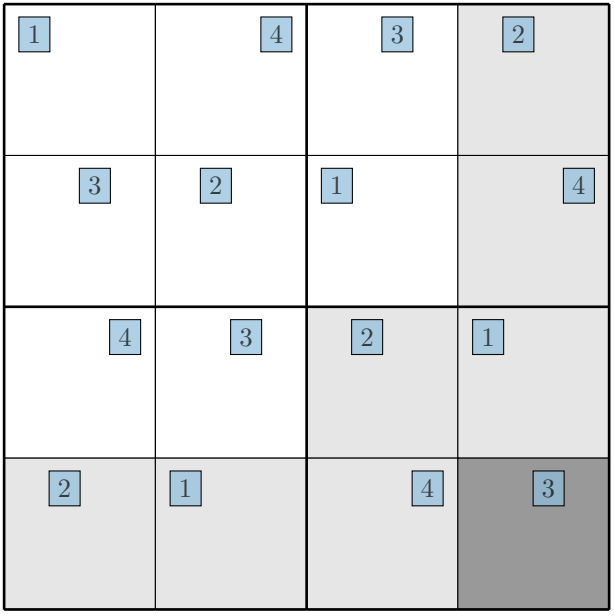


`solve`(`problem`: unary constraints)

- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict



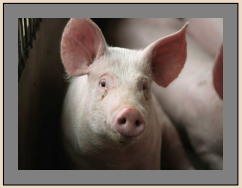
`solve`(`problem`: unary constraints)

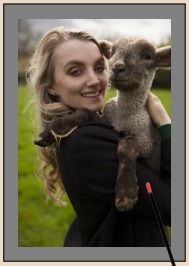
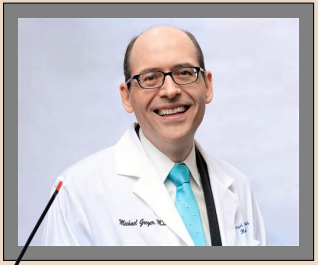
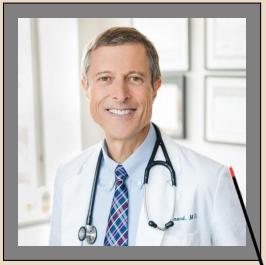
- `domains` = `init_domains`()
- `restrict_domains`(`problem`, `domains`)
- initialize `stack` as an empty stack
- while `true`:
  - if `propagate`(`domains`) reports no conflict:
    - if |`domains`[`spot`]| = 1 for all `spots`, this is a consistent assigment
    - `orig_domains` = `copy`(`domains`)
    - `spot`, `num` = `search`(`domains`)
    - add (`spot`, `num`, `orig_domains`) to `stack`
  - otherwise:
    - if `stack` is empty, there is no consistent assignment
    - `domains` = `backtrack`(`stack`)

`propagate`(`domains`: tile domains)

- do until `domains` doesn't change:
  - for each `spot` on the board:
    - if |`domains`[`spot`]| = 1, remove `spot`'s `num` from its `peers`' domains
    - if the above caused any domain to be empty, there is a conflict




SUPREME BENEVOLENT