Satisfiability Checking

Branch and Bound

Prof. Dr. Erika Ábrahám

Theory of Hybrid Systems
Informatik 2

WS 10/11

Integer Linear Systems

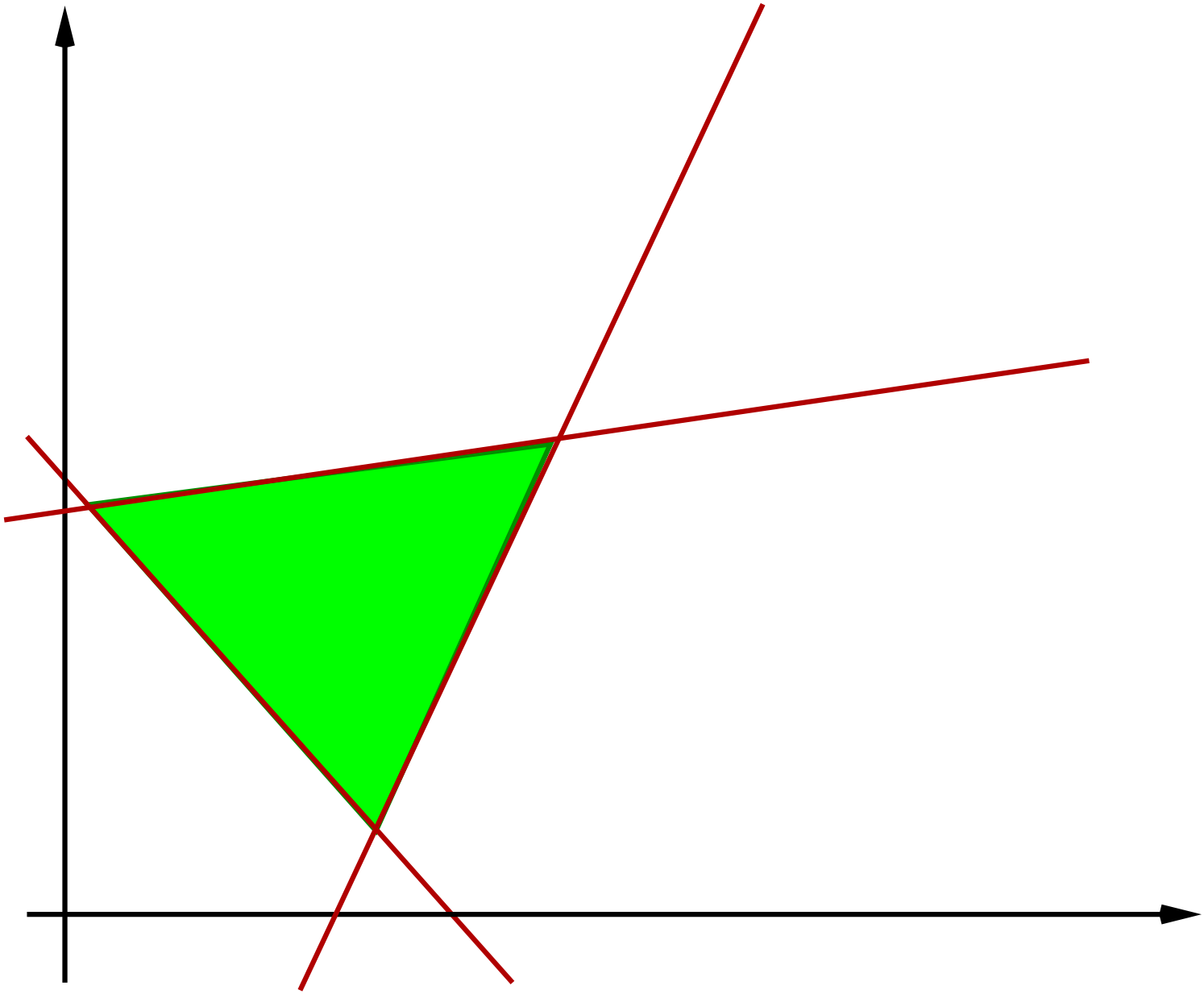
Definition

An *integer linear system* S is a linear system $Ax = 0$, $\bigwedge_{i=1}^{m} l_i \leq s_i \leq u_i$, with the additional *integrality requirement* that all variables are of type integer.

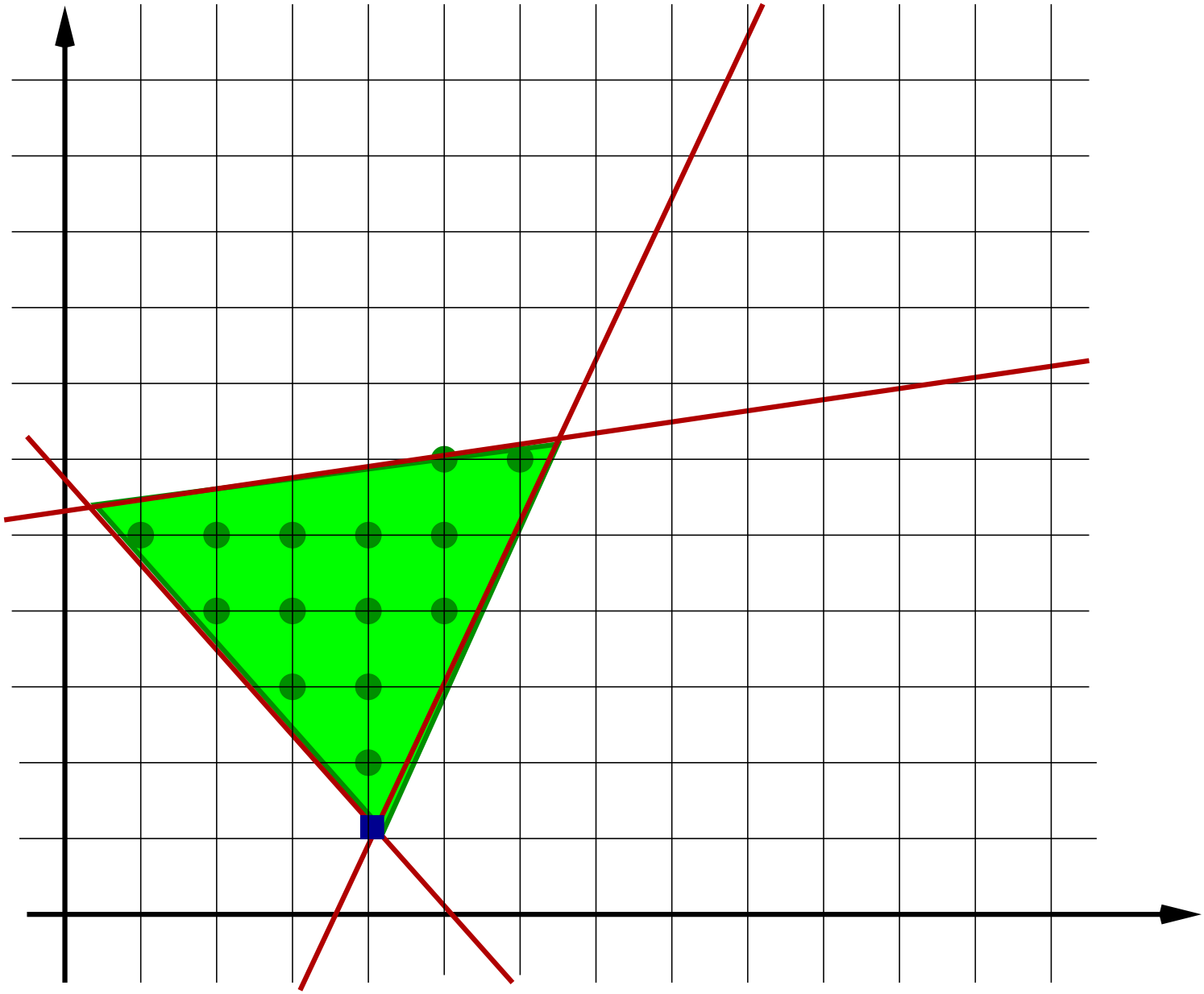
Definition (relaxed system)

Given an integer linear system S, its *relaxation relaxed(S)* is S without the integrality requirement.

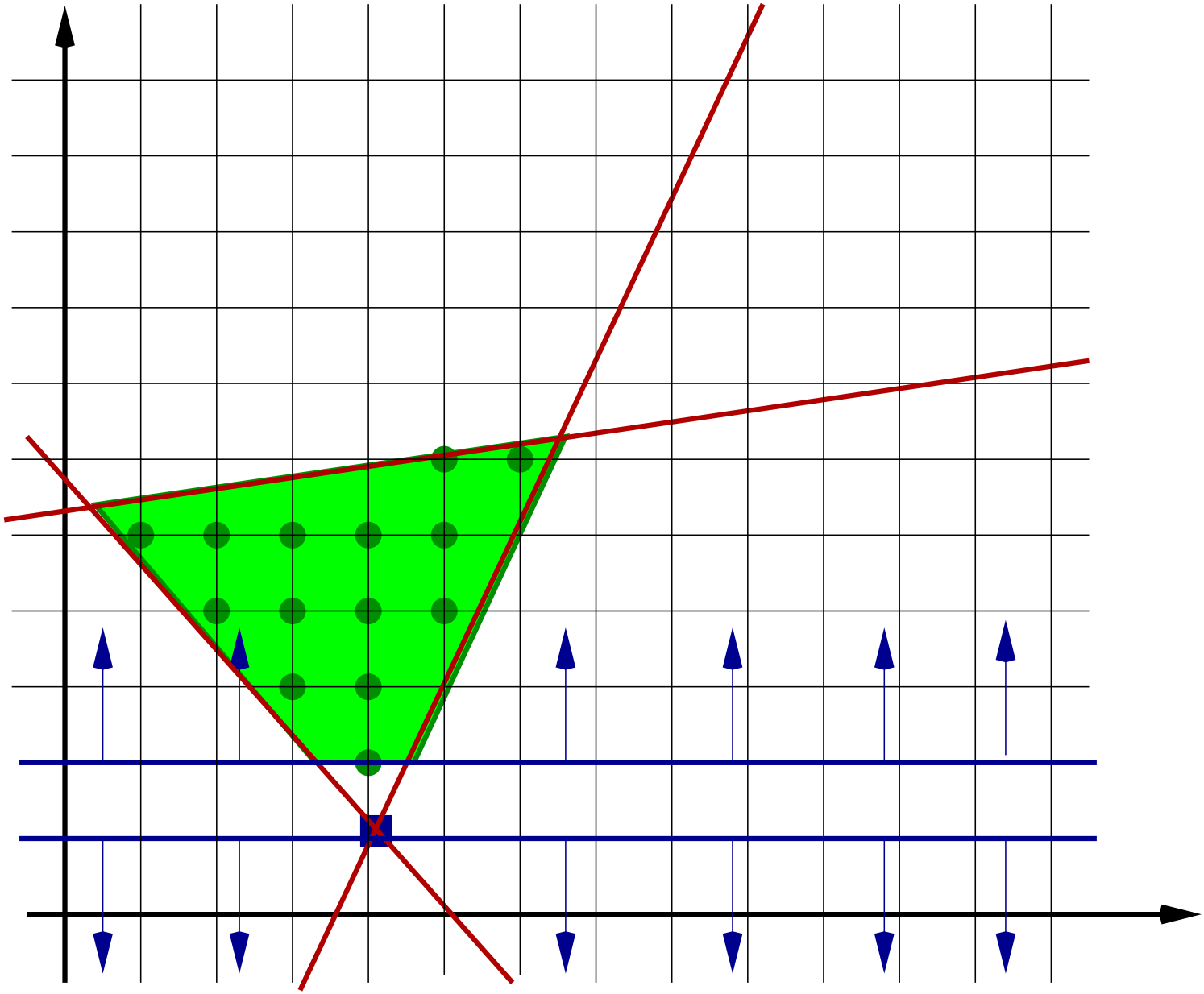
Geometrical Interpretation



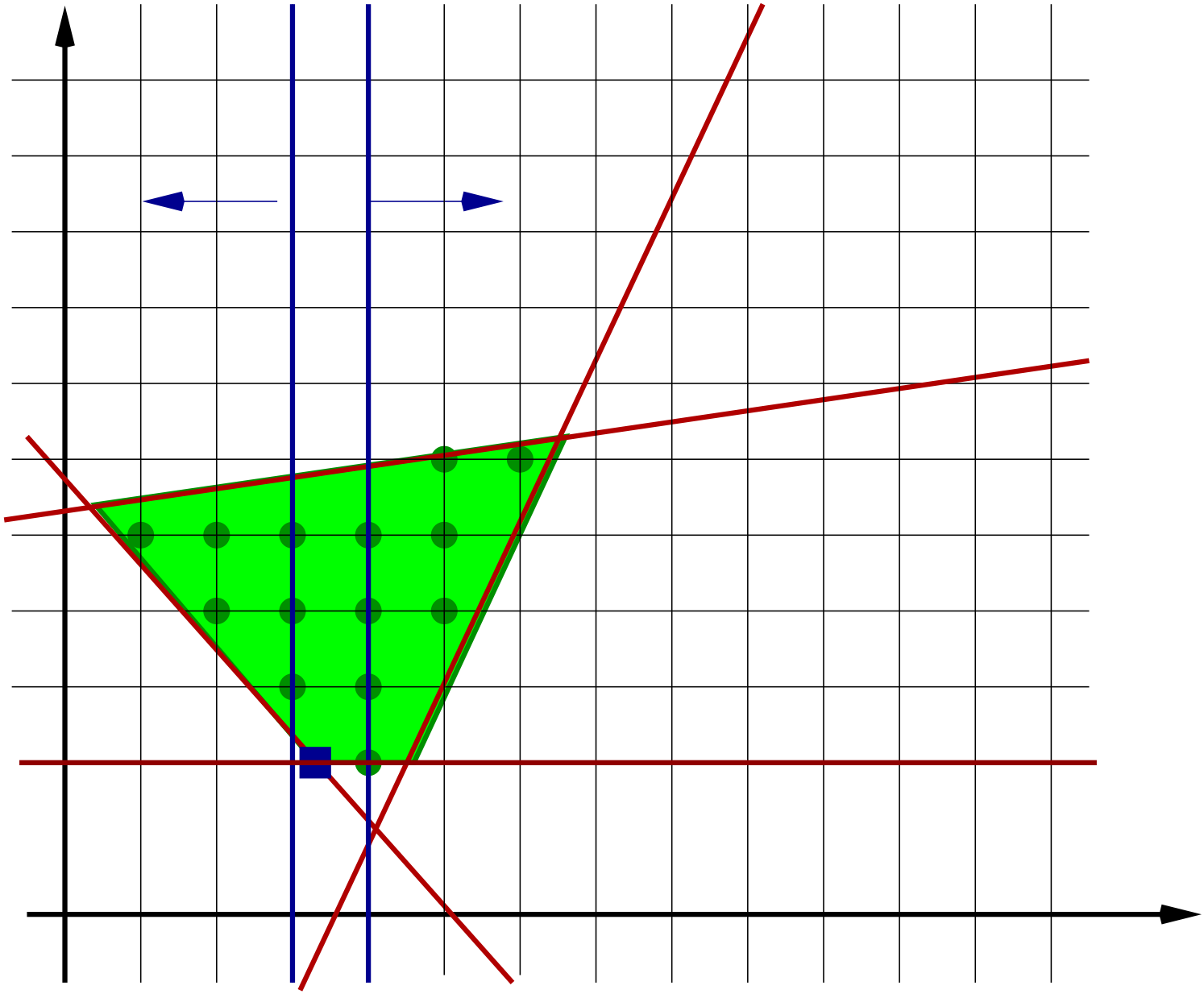
Geometrical Interpretation



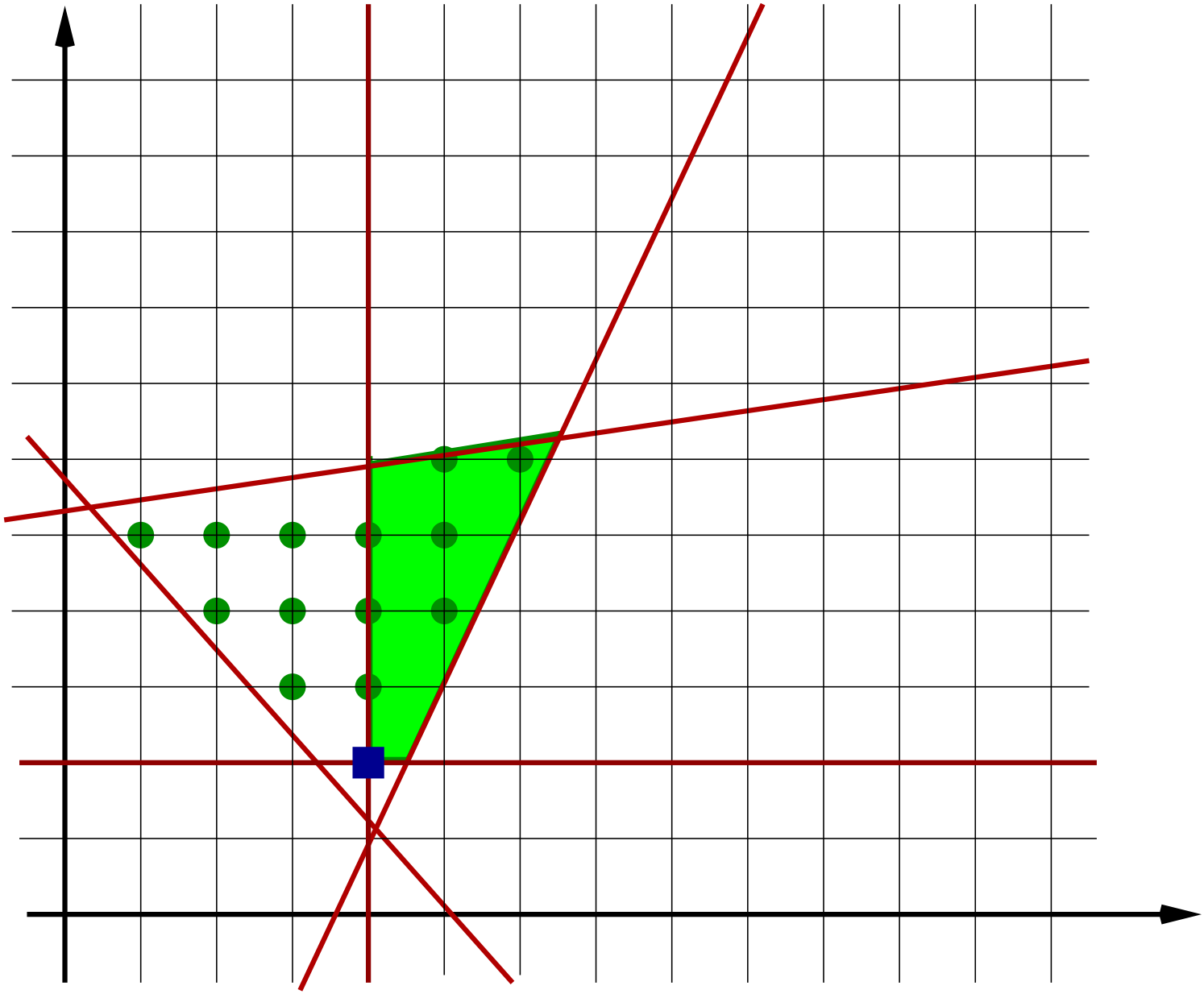
Geometrical Interpretation



Geometrical Interpretation



Geometrical Interpretation



Branch and Bound Algorithm

Input: An integer linear system S
Output: SAT if S is satisfiable, UNSAT otherwise

```
procedure Branch-and-Bound(S) {
  res = LP(relaxed(S));
  if (res==UNSAT) return UNSAT;
  else if (res is integral) return SAT;
  else {
    Select a variable v that is assigned a nonintegral value r;
    if (Branch-and-Bound(S ∪ (v ≤ ⌊r⌋))==SAT) return SAT;
    else if (Branch-and-Bound(S ∪ (v ≥ ⌈r⌉))==SAT) return SAT;
    else return UNSAT;
  }
}
```

Branch and Bound

- The algorithm is incomplete.
- Example: $1 \leq 3x - 3y \leq 2$ has unbounded real solutions but no integer solutions $\rightarrow$ the algorithm loops forever.
- The algorithm can be made complete for formulae with the small-model property: if there is a solution, then there is also a solution within a (computable) finite bound.
- The algorithm can be extended to *mixed integer programming*, where some of the variables are integers while the others are real.

Branch and Bound

- Branch: Split the search space
- Bound: Exclude unsatisfiable sub-spaces
- We have seen: Depth-first search
- Also possible: Breadth-first search

Branch and Bound

Optimizations:

- Constraints can be removed: $x_1 + x_2 \leq 2$, $x_1 \leq 1$, $x_2 \leq 1$.
  First constraint is redundant.
- Bounds can be tightened: $2x_1 + x_2 \leq 2$, $x_2 \geq 4$, $x_1 \leq 3$
  From the first two constraints we get $x_1 \leq -1$
- Assume a constraint $\sum_i a_i x_i \leq b$ with $l_i \leq x_i \leq u_i$.

  If $a_k > 0$, we have $x_k \leq (b - \sum_{i \neq k} a_i l_i)/a_k$.

  If $a_k < 0$, we have $x_k \geq (b - \sum_{i \neq k} a_i u_i)/a_k$.

Cutting planes

- We looked at branching by dividing the value domain of an integer variable into two halfs.
- We could also cut with other, better contraints.
- E.g., for $x \in \mathcal{Z}$, from $2x \leq 11$ we can conclude $x \leq 5$.
- But how to generate such cutting planes?
- We look at one method for generating cutting planes: Gomory's cuts.

Gomory's cuts

As before, use Simplex to find a real solution. We generate a new constraint such that the new (reduced) feasible region has two important properties:

- It does not contain the solution to the old LP problem obtained by the Simplex method.
- It contains all feasible solutions to the original ILP problem. That is, the cut does not remove any feasible solution of the original integer programming problem.

Gomory's cuts

Example:

- After finding an LP-solution, assume that a row of the tableau represents the constraint
  $3x_1 + 5/4x_2 + 1/3x_3 = 8/3.$
- We rewrite this constraint by separating each of the coefficients into two parts: an integer part and a fraction. The fraction is required to be non-negative.
  $(3 + 0/1)x_1 + (1 + 1/4)x_2 + (0 + 1/3)x_3 = 2 + 2/3.$
- Next, we collect all the integer parts (including the integer part of the RHS) on the left hand side of the equation.
  $0/1x_1 + 1/4x_2 + 1/3x_3 + (3x_1 + x_2 + 0x_3 - 2) = 2/3$
- The red term is smaller or equal zero. Thus we can conclude
  $0/1x_1 + 1/4x_2 + 1/3x_3 >= 2/3$