Solving two-player games

We are considering two approaches for encoding the chess challenge domain as a ***general reasoning task***:

A) Propositional satisfiability (SAT) encoding & SAT solvers:

- pros: state-of-the-art, very sophiscated solvers
- cons: very large SAT encodings

B) Quantified Boolean Formulae (QBF) & QBF solvers:

- pros: much smaller encodings, better scaling than SAT approach.
- cons: relatively new field, solvers not yet sophisticated enough, but show promise.

We have focused on option B.

Main result: A new QBF encoding strategy coupled with our new QBF solver gives over five orders of magnitude speed-up. Shows promise for scalability of general reasoning to chess challenge domain!

QBF encodings

- Case study: chess instances (capture black king in k moves)
  - white (black) player is the existential (universal) player.
- **Variables**:
  - L_i: set of Boolean variables representing location of pieces at step i
  - $M_i^w (M_i^b)$: set of Boolean variables representing moves of white (black) at step i
- **Clauses** (transition axioms):
  - I: set of clauses representing the initial location of pieces at step 0
  - $T_i^w (T_i^b)$: set of clauses representing transition axioms for white (black) player; preconditions, moving effects and frame axioms at step i (see SatPlan Kautz & Selman '96, '04)
  - G: set of clauses representing the goal: "the black king is not on board at step k+1 and the white king is on board at each step"
- **Formula (I)** :

$$\exists M_0^w \forall M_1^b \mathsf{K}\ \exists M_{k-2}^w \forall M_{k-1}^b \exists M_k^w \exists L_0 \mathsf{K}\ L_{k+1}$$

$$I \wedge (T_0^w \wedge \mathsf{K}\ \wedge T_k^w) \wedge (T_1^b \wedge \mathsf{K}\ \wedge T_{k-1}^b) \wedge G$$

Are we done?

What about illegal moves?

Encodings, cont.

Informally, formula (I) says,

White does whatever it can to satisfy all clauses,

while Black will try to falsify at least one, i.e.:

Does there exist a 1st move for White, such that

for all possible 1st moves for Black, such that

there exists a 2nd move for White, such that

for all possible 2nd moves for Black, such that

...

[the set of logical clauses encoding

"Black king captured" is satisfied.]

Counter-intuitive issue:

Black can simply falsify this expression by making an "illegal" move, violating the transition axioms, e.g., moving a piece off the board or jumping over a piece, etc.! We have to prevent this in our encoding.

Capturing Illegal Moves

Illegal move: an action that violates the rules of the game

"illegal moves causes formula (I) to be false"

White has no incentive to make illegal moves; but Black does!

We identify two types of illegal moves for black player; static and dynamic

1) Static illegal moves:

ex: Black player moves two pieces at a given step

idea: introduce indicator variables that force the formula to be true whenever black player does not perform exactly one move

ilm_i: indicator variable for static illegal moves at step i, z: global indicator variable

2) Dynamic illegal moves, due to the violation of the precondition of moves:

ex: black player moves to a forbidden cell.

(I) black only falsifies the formula if its transitions are true and the goal is false,

(II) introduce indicator variables for dynamic illegal moves, as in static illegal moves.

Capturing Illegal Moves, cont.

We obtain formula (II):

$$\exists M_0^w \forall M_1^b \mathsf{K}\ \exists M_{k-2}^w \forall M_{k-1}^b \exists M_k^w \exists L_0 \mathsf{K}\ L_{k+1} \exists ilm_1 \mathsf{K}\ ilm_{k-1} \exists z$$

$$(((I \wedge (T_0^w \wedge \mathsf{K}\ \wedge T_k^w) \wedge (T_1^b \wedge \mathsf{K}\ \wedge T_{k-1}^b) \wedge G) \vee z) \wedge (z \Leftrightarrow ilm_1 \vee \mathsf{K}\ \vee ilm_{k-1}))$$

This is a logically sound and complete encoding.

Next issue: computational efficiency and scalability.

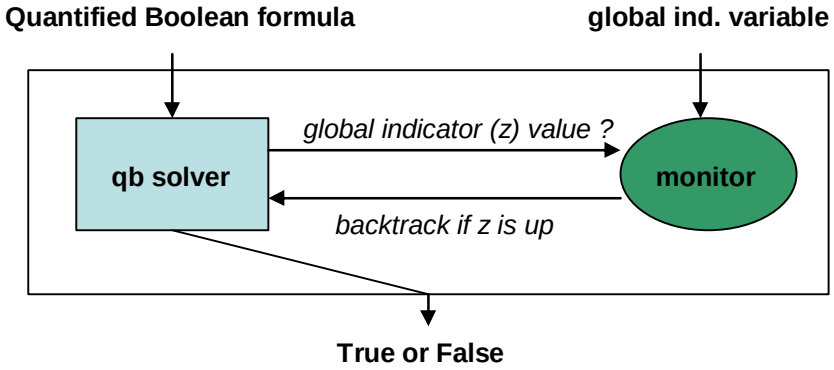
Conditional QBF

Standard QBF solvers do not scale well on the QBF encoding because propagation mechanisms cannot handle the indicator variables efficiently.

We introduced a new approach, which we refer to as “conditional QBF” formulation and solver:

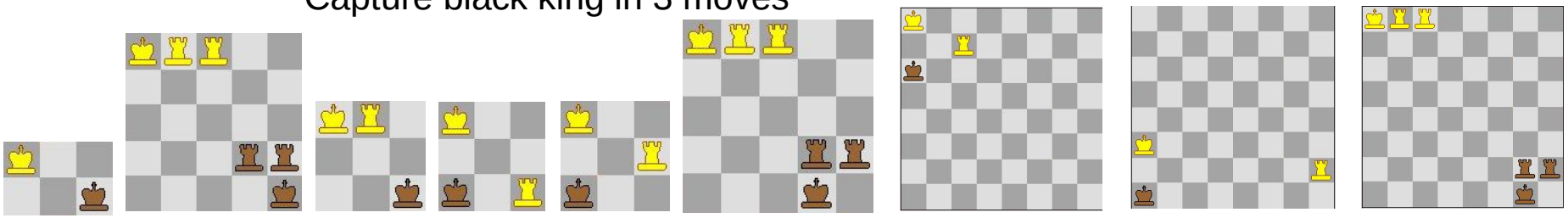
- take the formula with the indicator variables.
- monitor search under the following condition:

“if the global indicator (z) is up, backtrack.
Otherwise, proceed as usual”



First Experimental results

"Capture black king in 3 moves"



1 unsat	2 unsat	3 sat	4 sat	5 sat	6 sat	7 unsat (see ex. 5)	8 sat (see ex. 4)	9 unsat (see ex. 6)
415 v.	151118 v.	1198 v.	1198 v.	1198 v.	151118 v.	92908 v.	92908 v.	2458220 v.
2055 cl.	908219 cl.	10698 cl.	10698 cl.	10698 cl.	908219 cl.	21118333 cl.	21118333 cl.	14765789 cl.

Performance of QBF solvers

	Non Conditional			Conditional
instance	quaffle	semprop	qube	cquaffle
1	3708	0.01	0.01	0.01
2	-	*	133	9
3	-	-	-	0.01
4	-	-	-	0.02
5	-	-	-	0.01
6	-	*	-	9
7	-	*	*	3.5
8	-	*	*	5.12
9	*	*	*	*

- QBF solvers: quaffle, semprop and qube (qube-rel1.3)
- Conditional QBF solver: cquaffle (Cornell) "extension of solver quaffle (Princeton)"

Conditional approach speeds up QBF solver by over five orders of magnitude !!!

Time (secs): '-' did not complete in 20,000 seconds; '*' formula too large to execute

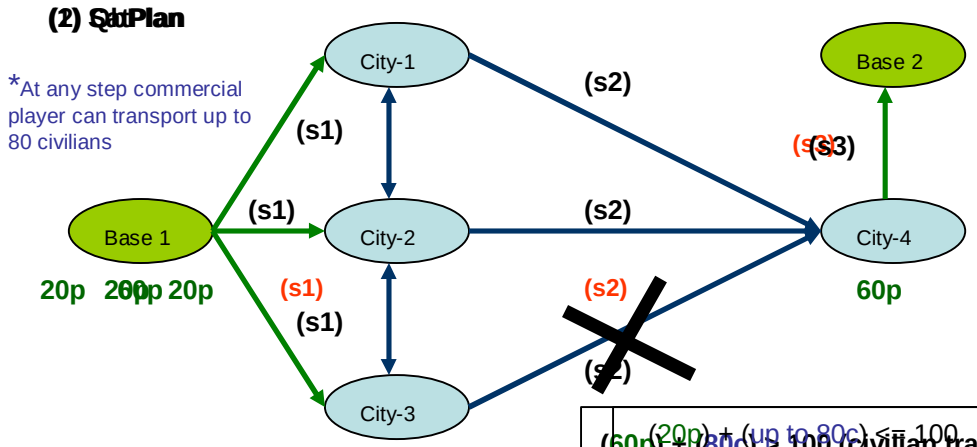
To more general queries

- (ii) To more general queries (related to military interests):
  - General scheme: objective under condition C. (where objective = Mate in L steps)
    1. Condition: certain white pieces have to be at certain locations at certain steps.
    2. Condition: certain white pieces must remain on board.
    3. Condition: certain white pieces can not move (attack) before other white pieces reach a certain location.
  - Multi-Objectives: (objective1 or objective2 or …) under condition C. (where objective = disjunction of intercahngeable targets)
    - Ex. ((capture b.queen at step L) or (capture b.rook and b.bishop before step L)) under condition C.

From Chess to Logistics Problems

- Classical planning addresses only tasks independent of any adversarial actions or unexpected failures.
  - replanning required or plan repair
  - response time of system at operation time, and basic efficiency of plans can be seriously degraded
- In order to address the problem, the planning community has carried several attempts to develop contingent planners
  - A contingent plan has several alternative options for each time stamp. The choice of the action is conditioned by the state of the environment at the execution time (“close to Chess description”)
  - However, no domain-independent technique for contingent planning has been developed until these days
- The principles of the conditional QB approach for Chess can be reused “in a straightforward manner” for any problem of deterministic adversarial planning
  - Deterministic classical planning can be solved successfully by applying SatPlan
  - We propose to solve deterministic adversarial planning by applying “Contingent” QB-PLAN

Example of Logistics



- **Blue nodes are cities, green nodes are military bases**
- **Blue edges are commercial transports, green edges are military**
- **Green edges (transports) have a capacity of 60 people, blue edges have a capacity of 100 people**
- **operator: "transport" t(who, amount, from, to, step)**
- **parallel actions can be taken at each step**
- **Goal: Send 60 personal from Base-1 to Base-2 in at most 3 steps**

(60p) + (80c) > 100 (civilian transport capacity)	(20p) + (up to 80c) <= 100	**Replan needed !!!**

- **One player: military player, deterministic classic planning, SatPlan**
  - (1) Sat-Plan: **t(m, 60, base-1, city-3, 1), t(m, 60, city-3, city-4, 2), t(m, 60, city-4, base-2, 3)**
- **Two players: deterministic adversarial planning QB Plan**
  - Military Player (m) is "white player", Commercial Player (c) is "black player" (Chess analogy). Commercial player can move up to **80 civilians** between cities. Commercial moves can not be invalidated. Goal can be rereaded as: "**send 60 personal from Base-1 to Base-2 in at most 3 steps whatever commercial needs (moves) are**"
  - If commercial player decides to move 80 civilians from city-3 to city-4 at the second step, we should replan (1). Indeed, the goal can not be achieved if we have already taken the first action of (1)
  - (2) QB-Plan: **t(m, 20, base-1, city-1, 1), t(m, 20, base-1, city-2, 1), t(m, 20, base-1, city-3, 1), t(m, 20, city-1, city-4, 2), t(m, 20, city-2, city-4, 2), t(m, 20, city-3, city-4, 2), t(m, 60, city-4, base-2, 3)**