CS 228 Assignment 2

Aradhana R (24b1006) Suhas Alladaboina (24b1009)

October 4, 2025

Contents

1 Perfect and doomed schemas for $n = 4$

Roll-derived challenges. From the roll numbers 1009 and 1006 we set

$$R_1 = \big(\{a_1, b_1\}, \{c_1, d_1\}\big) = (\{1, 0\}, \{0, 9\}), \qquad R_2 = (\{1, 0\}, \{0, 6\}).$$

For the two remaining challenges we choose

$$R_3 = (\{0, 0\}, \{0, 1\}), \qquad R_4 = (\{0, 0\}, \{0, 2\}).$$

Thus $a_3 = b_3 = 0$, $c_3 = 0$, $d_3 = 1$ and $a_4 = b_4 = 0$, $c_4 = 0$, $d_4 = 2$

1.1 A : A perfect schema for $\Theta = 2$

Because $a_3 = b_3 = 0$ and $a_4 = b_4 = 0$, the Keeper's choices in rounds 3 and 4 do not change the sum. It therefore suffices to branch based only on the first two Keeper choices. We write the schema $s = (s_1, s_2, s_3, s_4)$ explicitly as:

- $s_1 : \{a_1, b_1\} \to \{c_1, d_1\}$ with $s_1(*) = c_1$ (i.e., always cast the 0-spell)
- $s_2 : \{a_1, b_1\} \times \{a_2, b_2\} \to \{c_2, d_2\}$ with $s_2(*, *) = c_2$ (again, always cast 0).
- $s_3 : \{a_1, b_1\} \times \{a_2, b_2\} \times \{a_3, b_3\} \to \{c_3, d_3\}$ given by

$$s_3(x_1, x_2, x_3) = \begin{cases} d_3\ (= 1), & \text{if } x_1 + x_2 \neq 0 \quad \& \quad x_1 \cdot x_2 = 0 \\ c_3\ (= 0), & \text{otherwise} \end{cases}$$

- $s_4 : \{a_1, b_1\} \times \{a_2, b_2\} \times \{a_3, b_3\} \times \{a_4, b_4\} \to \{c_4, d_4\}$ given by

$$s_4(x_1, x_2, x_3, x_4) = \begin{cases} d_4\ (= 2), & \text{if } x_1 = 0 \quad \& \quad x_2 = 0 \\ c_4\ (= 0), & \text{otherwise} \end{cases}$$

Correctness (perfectness): There are only four distinct Keeper-contribution patterns because rounds $3, 4$ always add 0:

$$(1, 1, 0, 0),\ (1, 0, 0, 0),\ (0, 1, 0, 0),\ (0, 0, 0, 0)$$

By direct evaluation of doors+spells prescribed above we get

Keeper doors	Door sum	Spells (by s)	Spell sum
$(1, 1, 0, 0)$	2	$(0, 0, 0, 0)$	0
$(1, 0, 0, 0)$	1	$(0, 0, 1, 0)$	1
$(0, 1, 0, 0)$	1	$(0, 0, 1, 0)$	1
$(0, 0, 0, 0)$	0	$(0, 0, 0, 2)$	2

In every case the total equals $\Theta(= 2)$. Hence every journey conforming to s yields the target power, so s is perfect.

1.2 B : A doomed schema for $\Theta = 1000$

With the same R_1, R_2, R_3, R_4 as above. For any journey the door sum is at most $1 + 1 = 2$ and the total spell sum is at most $9 + 6 + 1 + 2 = 18$. Thus the maximum achievable power is $2 + 18 = 20 < 1000$. Therefore no schema can meet $\Theta = 1000$ for all journeys, so every schema is doomed for this instance.

An explicit doomed schema. Define a schema $s = (s_1, s_2, s_3, s_4)$ that *always* casts the 0-spell in every round:

- $s_1 : \{a_1, b_1\} \to \{c_1, d_1\}$ with $s_1(*) = c_1$
- $s_2 : \{a_1, b_1\} \times \{a_2, b_2\} \to \{c_2, d_2\}$ with $s_2(*, *) = c_2$
- $s_3 : \{a_1, b_1\} \times \{a_2, b_2\} \times \{a_3, b_3\} \to \{c_3, d_3\}$ given by $s_3(*, *, *) = c_3$
- $s_4 : \{a_1, b_1\} \times \{a_2, b_2\} \times \{a_3, b_3\} \times \{a_4, b_4\} \to \{c_4, d_4\}$ given by $s_4(*, *, *, *) = c_4$

Under this schema, when the keeper's choice is (b_1, b_2, b_3, b_4) the spell sum and the keeper sun are both 0 and $0 + 0 \neq 1000$, hence we have shown a confirming journey which does not yeild target power therefore this is a doomed schema.

2 Deciding T_ζ-satisfiability in $O(n^k \cdot \text{length}(\varphi))$ time

Input. A fixed $\zeta = \langle [n], \mathcal{P}, \Lambda \rangle$ and a formula $\varphi = Q_1 x_1 \cdots Q_k x_k \, \psi$, where ψ is an elementary sub-formula.

2.1 Algorithm

We evaluate the quantifier prefix by recursive descent. When substituting $a \in [n]$ for a variable, we immediately simplify each atomic $P(x_i)$ to $\top$ or $\bot$ according to Λ and then evaluate the resulting propositional formula.

Algorithm 1 TSAT(φ)

```
function CHECK(φ)
    if φ has no quantifiers then
        return Value(φ)                    ▷ propositional evaluation after all substitutions
    else if the first quantifier is ∃x then
        Let φ = ∃xϕ
        return CHECK_∃(∃xϕ)
    else
        Let φ = ∀xϕ
        return CHECK_∀(∀xϕ)

function CHECK_∃(∃x ϕ)
    for each a ∈ [n] do
        if CHECK(ϕ[a/x]) then
            return true
    return false

function CHECK_∀(∀x ϕ)
    for each a ∈ [n] do
        if not CHECK(ϕ[a/x]) then
            return false
    return true

return SAT if CHECK(φ) else UNSAT
```

2.2 Correctness

We prove by induction on the number of quantified variables that CHECK(φ) returns *true* iff $\models_\zeta \varphi$.

Base Case($k = 0$): $\varphi = \psi$ has no quantifiers. By semantics, $\models_\zeta \psi$ iff Value(ψ) obtained after simplifying every atomic $P(t)$ using Λ is $\top$.

Induction step: Write $\varphi = Qx\,\phi$ and assume the claim holds for all formulas with fewer quantifiers.

- If $Q = \exists$, by semantics, $\models_\zeta \varphi$ iff there exists $a \in [n]$ with $\models_\zeta \phi[a/x]$. By the IH, this holds iff CHECK($\phi[a/x]$) returns **true** for some a, which is what CHECK_$\exists$ checks.
- If $Q = \forall$, by semantics, $\models_\zeta \varphi$ iff for all $a \in [n]$, $\models_\zeta \phi[a/x]$. By the IH, this holds iff CHECK($\phi[a/x]$) returns **true** for every a, which is what CHECK_$\forall$ checks.

Thus the algorithm is correct.

2.3 Time complexity

Let k be the number of quantified variables, set $n \coloneqq |[n]|$. The running time T_k where $k \geq 1$ satisfies the recurrence $T_k = n\,T_{k-1} + c_k$ where c_k is $O(1)$ because each quantifier tries all the n elements and makes a recursive call on a shorter prefix and propositional evaluation of ψ takes time $O(\text{length}(\varphi))$.

$$\begin{aligned} T_0 &= O(\text{length}(\varphi)) \\ T_k &= n\,T_{k-1} + c_k \qquad (k \geq 1), \end{aligned}$$

$$T_k = n^k T_0 + \sum_{i=0}^{k-1} n^i\, c_{k-i} \;\subseteq\; n^k T_0 + O\left(\sum_{i=0}^{k-1} n^i\right) = n^k T_0 + O(n^k)$$

$$T_k = O(n^k\,\text{length}(\varphi))$$

2.4 Space complexity

We run the recursion depth-first and the maximum stack depth is k and we need only $O(1)$ extra space per frame (the current index a and a pointer to the current subformula). Since the search is depth-first, we will never have two sibling nodes active at the same time, and therefore the space used is bounded by the maximum depth rather than the total size of the tree. The propositional formula φ itself is stored externally and has size length(φ) (in the tree we only keep pointers into this formula). Thus the total space used is polynomial: $O(k + \text{length}(\varphi))$.

3 Reduction of the schema-existence problem to T_ζ-satisfiability

3.1 Fixed structure

We fix the structure

$$\zeta^\star = \langle [2],\, \{T\},\, \Lambda \rangle, \qquad \Lambda(T) = \{1\}.$$

Thus a term u is either 0 or 1 and the only unary predicate $T(u)$ is true iff $u = 1$. Intuitively $T(u)$ is the bool cast of u.

3.2 Encoding bits and conditionals

For formulae A, B, C and term u, define macros:

$$A \oplus B \equiv \mathrm{XOR}(A,B) \equiv \big((A)\vee(B)\big)\wedge\neg\big((A)\wedge(B)\big), \qquad \mathrm{ITE}(u,A,B) \equiv \big((T(u))\wedge(A)\big) \vee \big(\neg(T(u))\wedge(B)\big)$$

$$\mathrm{MAJ}(A,B,C) \equiv \big((A)\wedge(B)\big) \;\vee\; \big((A)\wedge(C)\big) \;\vee\; \big((B)\wedge(C)\big).$$

Intuitively $\mathrm{ITE}(u,A,B)$ selects A if $u=1$ and B if $u=0$.

3.3 Contributions of each round

Let $\Omega = (R_1,\ldots,R_n)$ with $R_i = (\{a_i,b_i\},\{c_i,d_i\})$ and a target Θ. Let $L \geq 1$ be any number of bits such that all numbers involved fit into L bits (e.g., $L = 2 + 2\lceil \log_2(n\cdot\Theta + \sum_i (a_i+b_i+c_i+d_i))\rceil$). For $t = 0,\ldots,L-1$ let $\mathrm{bit}_t(x)$ be the t-th binary bit of x.

For variable x_i (Keeper's choices) and variable y_i (Spellcaster's responses) define the t-th contributed bit in challenge i by

$$K_{i,t}(x_i) \;:=\; \mathrm{ITE}\big(x_i,\; T(\mathrm{bit}_t(a_i)),\; T(\mathrm{bit}_t(b_i))\big), \qquad S_{i,t}(y_i) \;:=\; \mathrm{ITE}\big(y_i,\; T(\mathrm{bit}_t(d_i)),\; T(\mathrm{bit}_t(c_i))\big).$$

Both are elementary subformulae over T.

3.4 A 2-input ripple-carry adder as formulas (bitwise full-adder)

Given formulas A, B, C, define the full-adder output macros

$$\mathrm{SUM}(A,B,C) \equiv (A \oplus B) \oplus C, \qquad \mathrm{CARRY}(A,B,C) \equiv \mathrm{MAJ}(A,B,C).$$

Now make a ripple carry adder for L bits by recursion:

$$\begin{aligned}
C_{-1}^{j} &:= \bot && 1 \leq j \leq 2n \\
S_t^0 &:= \bot && 0 \leq t < L \\
S_t^j &:= \mathrm{SUM}\big(S_t^{(j-1)},\; I_t^{(j)},\; C_{t-1}^{(j)}\big) && 0 \leq t < L,\; 1 \leq j \leq 2n \\
C_t^j &:= \mathrm{CARRY}\big(S_t^{(j-1)},\; I_t^{(j)},\; C_{t-1}^{(j)}\big) && 0 \leq t < L,\; 1 \leq j \leq 2n
\end{aligned}$$

where the I_i's are defined as

$$I_t^j := \begin{cases} K_{j,t}(x_j), & 1 \leq j \leq n, \\ S_{j-n,t}(y_{j-n}), & n < j \leq 2n. \end{cases}$$

Thus each S_t^j is a formula that can be uniquely determined from Ω and $x_1, x_2 \ldots x_n, y_1, y_2, \ldots y_n$.

3.5 Target equality

Let $\Theta = \sum_{t=0}^{L-1} \theta_t 2^t$ and put $\vartheta_t = T(\theta_t)$. The bitwise equality with the final sum is

$$\Psi_{\Omega,\Theta}(x_1,\ldots,x_n,y_1,\ldots,y_n) \;:=\; \bigwedge_{t=0}^{L-1} \big(S_t^{2n} \;\leftrightarrow\; \vartheta_t\big).$$

3.6 Final formula:

$$\boxed{\varphi_{\Omega,\Theta} \;:=\; \forall x_1\,\exists y_1\,\cdots\,\forall x_n\,\exists y_n\;\Psi_{\Omega,\Theta}.}$$

Everything inside $\Psi_{\Omega,\Theta}$ is built solely from $T(\cdot)$ by propositional connectives.

3.7 Correctness claim

For every trial (Ω, Θ) the student has a perfect incantation schema iff $\models_{\zeta^\star}\ \varphi_{\Omega,\Theta}$.

Lemma 1 (Ripple-adder correctness)**.** *Let z_t be the t-th output bit of the ripple adder (with carries c_t) when, in round i, the inputs to the t-th full adder are the bits $K_{i,t}$ and $S_{i,t}$. Let Θ be the target power and $(\theta_0, \ldots, \theta_{L-1})$ be the target bit vector i.e bit values of Θ and write*

$$\Psi_{\Omega,\Theta} \;:=\; \bigwedge_{t=0}^{L-1} (z_t \leftrightarrow T(\theta_t)).$$

Then, for every choice of input bits and initial carry $c_{-1} = 0$,

$$\Psi_{\Omega,\Theta} \quad \textit{holds} \quad \Longleftrightarrow \quad \textit{the binary sum equals } \Theta.$$

Proof: By the full-adder equations $z_t = K_{i,t} \oplus S_{i,t} \oplus C_{t-1}$ and $c_t = \mathrm{MAJ}(K_{i,t}, S_{i,t}, C_{t-1})$, an induction on t shows that the bit vector $(z_t)_{t=0}^{L-1}$ is exactly the binary representation of the integer sum of the inputs. Hence $\forall t\, z_t = T(\theta_t)$ iff the computed sum equals Θ.

Theorem 1 (Correctness claim)**.** $\models_{\zeta^\star}\ \varphi_{\Omega,\Theta}$ *iff there exists a perfect schema s.*

Proof. First, reduce to the adder. By the lemma, $\Psi_{\Omega,\Theta}$ holds iff the computed sum equals Θ.

($\Rightarrow$) Forward direction. Let $s = (s_1, \ldots, s_n)$ be a perfect schema, where each s_i maps Keeper prefixes $(x_1, \ldots, x_i)$ to $\{c_i, d_i\}$. We show that $\forall x_1\,\exists y_1\,\forall x_2\,\exists y_2\,\cdots\,\forall x_n\,\exists y_n\;\Psi_{\Omega,\Theta}$ holds in $\zeta^\star$.

Fix an *arbitrary* x_1; define $y_1 := 1$ iff s_1 prescribes d_1 on the prefix (x_1), and $y_1 := 0$ otherwise. Now, given an *arbitrary* x_2, define $y_2 := 1$ iff s_2 prescribes d_2 on (x_1, x_2); otherwise set $y_2 := 0$. Proceed inductively: for each round $i = 1, \ldots, n$, after an *arbitrary* Keeper move x_i, extend the current prefix, set $y_i := 1$ iff s_i prescribes d_i on current prefix, and 0 otherwise.

By construction, in every round the chosen $K_{i,t}$ and $S_{i,t}$ are exactly those picked by s on the realized prefix. Since s is perfect, the accumulated sum equals Θ for every Keeper play; by the Ripple–adder correctness lemma, this implies $\Psi_{\Omega,\Theta}$. Because the Keeper choices at each $\forall$-step were arbitrary and we produced matching witnesses at each $\exists$-step, the sentence $\varphi_{\Omega,\Theta}$ holds.

($\Leftarrow$) Backward direction. Assume $\models_{\zeta^\star}\ \varphi_{\Omega,\Theta}$. By the semantics of alternating quantifiers, there exist Skolem functions f_i from Keeper-prefixes to $\{0, 1\}$ such that setting $y_i = f_i(\text{prefix})$ ensures $\Psi_{\Omega,\Theta}$ for all combinations. Define a schema s by prescribing d_i when $f_i = 1$ and c_i otherwise. Then, for every Keeper play, the same choices make $\Psi_{\Omega,\Theta}$ hold. By the lemma, the computed sum is Θ, so the accumulated power is Θ and s is perfect.

This proves the equivalence. □

3.8 Pseudocode for mechanically emitting $\varphi_{\Omega,\Theta}$.

Algorithm 2 BUILD-FORMULA(Ω, Θ)

Input: $\Omega = ((\{a_i, b_i\}, \{c_i, d_i\}))_{i=1}^{n}$, target Θ
Choose $L \leftarrow 1 + \lceil \log_2(\Theta + \sum_i (a_i + b_i + c_i + d_i)) \rceil$
for $t = 0$ to $L - 1$ **do**
    $\theta_t \leftarrow \mathrm{bit}_t(\Theta)$
    for $i = 1$ to n **do**
        $K_{i,t}(x_i) \leftarrow \mathrm{ITE}(T(x_i), T(\mathrm{bit}_t(a_i)), T(\mathrm{bit}_t(b_i)))$ ▷ $K_{i,t}$ and $S_{i,t}$ are functions
        $S_{i,t}(y_i) \leftarrow \mathrm{ITE}\big(T(y_i), T(\mathrm{bit}_t(d_i)), T(\mathrm{bit}_t(c_i))\big)$ ▷ $\mathrm{bit}_t(\cdot)$ is computed mechanically.
for $t = 0$ to $L - 1$ **do**
    $S_t^0 \leftarrow \bot$
for $j = 1$ to $2n$ **do**
    $C_{-1}^j \leftarrow \bot$
for $j = 1$ to $2n$ **do**
    for $t = 0$ to $L - 1$ **do**
        $I_t^j \leftarrow \big(j \le n \ ?\ K_{j,t}(x_j)\ :\ S_{j-n,t}(y_{j-n})\big)$
        $S_t^j \leftarrow \mathrm{SUM}\big(S_t^{j-1}, I_t^j, C_{t-1}^j\big)$
        $C_t^j \leftarrow \mathrm{CARRY}\big(S_t^{j-1}, I_t^j, C_{t-1}^j\big)$ ▷ I_i^t,S_i^t and C_i^t are functions of x_j, y_{j-n}
$\Psi_{\Omega,\Theta} \leftarrow \bigwedge_{t=0}^{L-1} \big(S_t^{2n} \leftrightarrow T(\theta_t)\big)$
return $\varphi_{\Omega,\Theta} := (\forall x_1\, \exists y_1 \cdots \forall x_n\, \exists y_n)\ \Psi_{\Omega,\Theta}$

Size and time of construction: The final formula has size $O(Ln)$ (each of the $2n$ S_i contributes $O(L)$ connective occurrences. Time taken to build the formula is also $O(Ln)$.

3.9 Worked example

Trial (doors, spells)

$$R_1 = (\{1, 0\}, \{2, 0\}), \quad \Theta = 2.$$

Structure and atoms Work in $\zeta^\star = \langle [2], \{T\}, \Lambda \rangle$ with $\Lambda(T) = \{1\}$, so $T(1) = \top$ and $T(0) = \bot$.

Final formula $\varphi_{\Omega,\Theta} = \forall x_1\, \exists y_1\, \big(\neg T(x_1) \ \wedge\ \neg T(y_1)\big)$.

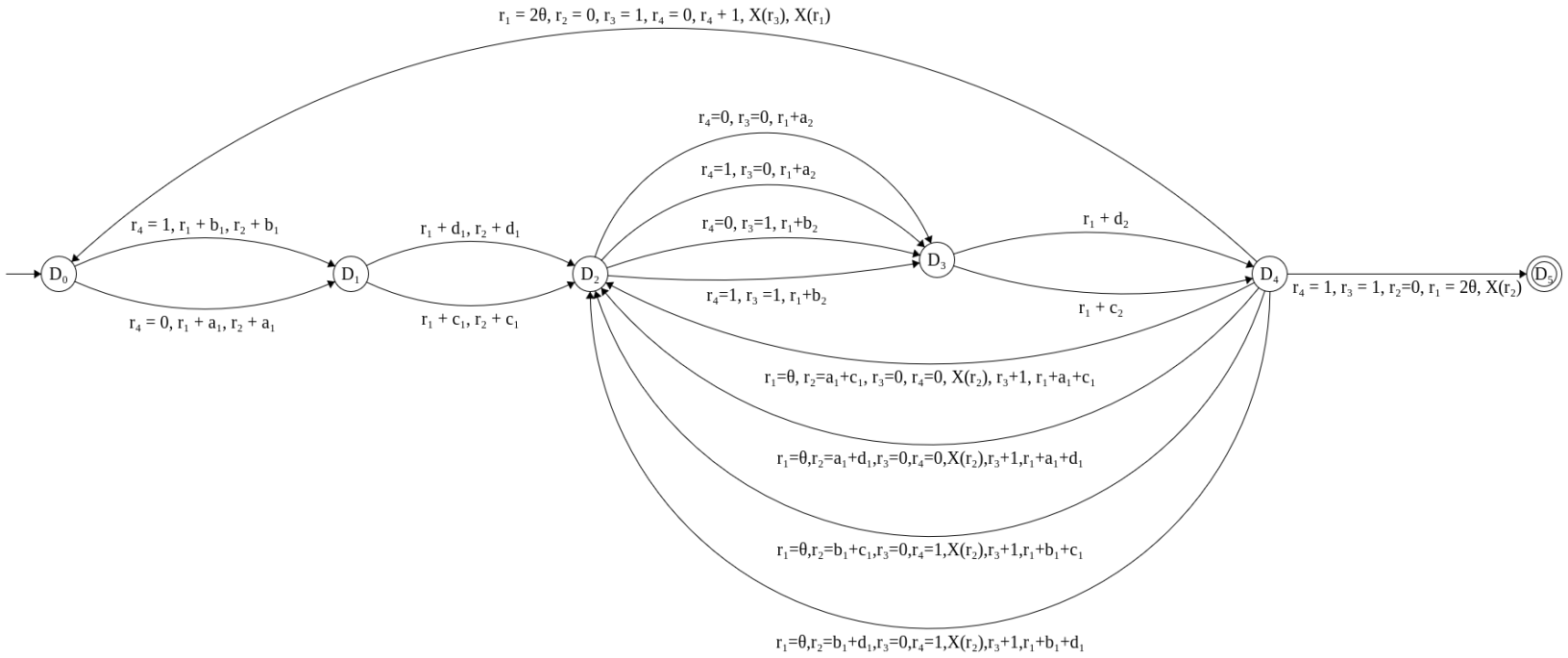
4 Reduction to D–Automata (two challenges)

Setting. A trial has two challenges $\Omega = (R_1, R_2)$ and a target integer $\Theta \in \mathbb{N}$.

4.1 Automaton idea

We simulate all four Keeper branches in series using 1 register to keep track of total sum, 1 register to keep track of the sum in first challenge, and 2 registers to keep track of branches we have covered out of 4 branches ($(x_1, x_2) \in \{a_1, b_1\} \times \{a_2, b_2\}$). In each branch, we non-deterministically pick the spell-caster's responses, in a way that depends on the Keeper prefix, and finally test that in each of the 4 branches the sums equal Θ. Reachability of the accepting configurations is equivalent to the existence of a perfect schema.

4.2 Automaton



Initial Configuration : $(D_0, r_1 = 0, r_2 = 0, r_3 = 0, r_4 = 0)$
Final Configuration : $(D_5, r_1 = 2\theta, r_2 = 0, r_3 = 1, r_4 = 1)$

4.3 Why the construction works (correctness)

Write $Reach(D_{\Omega,\Theta})$ for "The target configuration is reachable from the initial configuration in the constructed automaton $D_{\Omega,\Theta}$."

Let $\Omega = (R_1, R_2)$ with $R_1 = (\{a_1, b_1\}, \{c_1, d_1\})$ and $R_2 = (\{a_2, b_2\}, \{c_2, d_2\})$. A spell schema is a pair $s = (s_1, s_2)$ where

$$s_1 : \{a_1, b_1\} \to \{c_1, d_1\} \quad \text{and} \quad s_2 : \{a_1a_2, a_1b_2, b_1a_2, b_1b_2\} \to \{c_2, d_2\}$$

The schema s is *perfect* if for every Keeper choices $(x_1, x_2) \in \{a_1, b_1\} \times \{a_2, b_2\}$ the total $T(x_1, x_2) = x_1 + s_1(x_1) + x_2 + s_2(x_1x_2)$ equals the target Θ.

Our automaton $D_{\Omega,\Theta}$ starts at $(s_0, 0, \ldots, 0)$, uses all 4 deterministic Keeper-branches for a_i/b_i, and non-deterministic Spellcaster-branches for c_i/d_i, and has a running-sum register r_1 that adds the number on each chosen rune or spell.

($\Rightarrow$) Perfect schema $\to$ reachability: Assume there exists a perfect schema $s = (s_1, s_2)$. Then, for *each* Keeper pair $(x_1, x_2) \in \{a_1, b_1\} \times \{a_2, b_2\}$ there is a Spellcaster response $s_1(x_1) \in \{c_1, d_1\}$ and $s_2(x_1x_2) \in \{c_2, d_2\}$ such that $x_1 + s_1(x_1) + x_2 + s_2(x_1x_2) = \Theta$

Our automaton uses $r_3, r_4 \in \{0, 1\}$ as branch-index registers to simulate all four Keeper choices, value of r_4 determining Keeper's choice in challenge 1 and r_3 determining Keeper's choice in challenge 2, and $r_{\text{sum}} = r_1$ to accumulate the total and r_2 to accumulate the total in first challenge.

We show a run that starts at $D_0, 0, 0, 0, 0$ and reaches the final configuration $D_5, 2\Theta, 0, 1, 1$.

Lap 1 (We initially are at D_0 and have $r_1 = 0, r_2 = 0, r_3 = 0, r_4 = 0$): Starting at D_0 with $r_4 = 0$ will force us to go to D_1 and then we pick either c_1 or d_1 as dictated by our existing perfect schema i.e we pick $s_1(a_1)$ and reach D_2. After that we are forced down to D_3 since $r_4 = 0$, $r_3 = 0$ and then we pick $s_2(a_1, a_2)$ and reach D_4. Now since we know that $r_1 = a_1 + s_1(a_1) + a_2 + s_2(a_1, a_2) = \Theta$, we now reach D_2 with $r_1 = \theta + a_1 + s_1(a_1), r_2 = 0, r_3 = 1, r_4 = 0$.

Lap 2 (Continue with the same first Keeper choice, now take the second x_2-branch): We are at D_2 with $r_1 = \Theta + a_1 + s_1(a_1), r_2 = 0,\ r_3 = 1,\ r_4 = 0$. We are forced down to D_3 (since $r_4 = 0,\ r_3 = 1$) and now pick the round-2 spell $s_2(a_1, b_2)$, reaching D_4. By perfection of s we have $r_1 = \Theta + a_1 + s_1(a_1) + b_2 + s_2(a_1, b_2) = 2\Theta$. Hence we can take the success edge that closes the $x_1 = a_1$ group and prepares the second group. This moves us to D_0 with registers $r_1 = 0, r_2 = 0, r_3 = 0, r_4 = 1$.

Lap 3 (Start the second Keeper choice in challenge 1): At D_0 with $r_4 = 1$ will force us to go to D_1 and at D_1 with $r_4 = 1$ we take the $s_1(b_1)$ branch and reach D_2. Since $r_3 = 0, r_4 = 1$ we are forced to D_3 and now pick the round-2 spell $s_2(b_1, a_2)$, reaching D_4. Again, by perfection, $r_1 = b_1 + s_1(b_1) + a_2 + s_2(b_1, a_2) = \Theta$. We take the success edge for this sub-branch, which keeps us within the $x_1 = b_1$ group and sets up its second x_2-choice. We therefore return to D_2 with $r_1 = \Theta + b_1 + s_1(b_1), r_2 = 0, r_3 = 1, r_4 = 1$.

Lap 4 (Finish the second Keeper choice in challenge 2): From D_2 with $r_4 = 1,\ r_3 = 1$ we proceed to D_3, choose $s_2(b_1, b_2)$, and reach D_4. Perfection of schema results in $r_1 = 2\Theta$. Thus we take the final success edge that closes the $x_1 = b_1$ group and enters the accepting sink(D_5) with $r_1 = 2\Theta, r_2 = 0, r_3 = 1, r_4 = 1$. This completes the forward direction.

($\Leftarrow$) Reachability $\rightarrow$ perfect schema: Suppose $Reach(D_{\Omega,\Theta})$ holds, i.e., there exists a run from $(D_0, 0, 0, 0)$ to the final configuration $(D_5, 2\Theta, 0, 1, 1)$. By construction of our automata (by noting how the values of r_3, r_4 change we can show this), any successful run decomposes into four *laps* (segments) [Proof later]

$$\pi_1 : (D_0, 0, 0, 0, 0) \rightarrow (D_4, \Theta, c_1, 0, 0) \rightarrow (D_2, \Theta + c_1, 0, 1, 0),$$
$$\pi_2 : (D_2, \Theta + c_1, 0, 1, 0) \rightarrow (D_4, 2\Theta, 0, 1, 0) \rightarrow (D_0, 0, 0, 0, 1),$$
$$\pi_3 : (D_0, 0, 0, 0, 1) \rightarrow (D_4, \Theta, c_2, 0, 1) \rightarrow (D_2, \Theta + c_2, 0, 1, 1),$$
$$\pi_4 : (D_2, \Theta + c_2, 0, 1, 1) \rightarrow (D_4, 2\Theta, 0, 1, 1) \rightarrow (D_5, 2\Theta, 0, 1, 1)$$

Extracting a schema from the run: Read the run and construct the schema

- From the first lap:
  - In the first lap there must be a $D_1 - D_2$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_1, r_2 + c_1)$ or $(r_1 + d_1, r_2 + d_1)$ and assign it (c_1 or d_1) to $s_1(a_1)$.
  - In the same lap there must be a $D_3 - D_4$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_2)$ or $(r_1 + d_2)$ and assign it (c_2 or d_2) to $s_2(a_1, a_2)$.

- Since we have a check $r_1 = \Theta$ on $D_4 - D_2$ transition, it means that the partial construction of schema made till now is correct.

- From the second lap:
  - In this lap there must be a $D_3 - D_4$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_2)$ or $(r_1 + d_2)$ and assign it (c_2 or d_2) to $s_2(a_1, b_2)$.
  - Since we have a check $r_1 = 2\Theta$ on $D_4 - D_0$ transition, it means that the partial construction of schema made till now is correct.

- From the third lap:
  - In the third lap there must be a $D_1 - D_2$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_1, r_2 + c_1)$ or $(r_1 + d_1, r_2 + d_1)$ and assign it (c_1 or d_1) to $s_1(b_1)$.
  - In the same lap there must be a $D_3 - D_4$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_2)$ or $(r_1 + d_2)$ and assign it (c_2 or d_2) to $s_2(b_1, a_2)$.
  - Since we have a check $r_1 = \Theta$ on $D_4 - D_2$ transition, it means that the partial construction of schema made till now is correct.

- From the fourth lap:
  - In this lap there must be a $D_3 - D_4$ transition, check which branch the given run takes i.e check if the run goes through the transition labeled $(r_1 + c_2)$ or $(r_1 + d_2)$ and assign it (c_2 or d_2) to $s_2(b_1, b_2)$.
  - Since we have a check $r_1 = 2\Theta$ on $D_4 - D_0$ transition, it means that the partial construction of schema made till now is correct.

By construction, $s_1 : \{a_1, b_1\} \to \{c_1, d_1\}$ and $s_2 : \{a_1a_2, a_1b_2, b_1a_2, b_1b_2\} \to \{c_2, d_2\}$ are well defined and each branch sums to Θ.

The equalities above hold for all four Keeper choices $(x_1, x_2) \in \{a_1, b_1\} \times \{a_2, b_2\}$, so the schema $s = (s_1, s_2)$ constructed from the successful run guarantees the target sum Θ on every branch. Therefore s is a perfect schema.

Conclusion: We have shown both directions:

$$\text{Perfect schema for } (\Omega, \Theta) \;\leftrightarrow\; \mathit{Reach}(D_{\Omega,\Theta}).$$

Therefore the automaton construction is correct.

Proof of why any accepting run should divide into the above 4 segments: We start at $(D_0, r_1 = 0, r_2 = 0, r_3 = 0, r_4 = 0)$ and accept only at $(D_5, 2\Theta, 0, 1, 1)$. The flags change only on two edges: (i) $D_4 \to D_2$ sets $r_3 : 0 \to 1$ and (ii) $D_4 \to D_0$ sets $r_4 : 0 \to 1, r_3 : 1 \to 0$ (group switch). Since r_4 must end as 1 and can be set to 1 only by $D_4 \to D_0$, that edge must occur *exactly once* before the final $D_4 \to D_5$. Before taking $D_4 \to D_0$ we must complete the first x_1–group: its first subbranch uses $D_4 \to D_2$ (making $r_3 : 0 \to 1$), and its second subbranch returns to D_4 with $r_3 = 1$,

then takes $D_4 \to D_0$ (making $r_4 : 0 \to 1$, $r_3 := 0$). After the switch, the second x_1–group repeats the same two-subbranch pattern: first subbranch uses $D_4 \to D_2$ ($r_3 : 0 \to 1$), second subbranch ends at D_4 with $r_3 = 1$ and now $r_4 = 1$, enabling the final $D_4 \to D_5$. Hence every accepting run factors into exactly four segments (two subbranches per group).

5 Bonus: General reduction to D–Automata

6 Team contributions

- **Roll 1006:**
  - Built the perfect and doomed schema examples for $n = 4$ and verified totals against the target Θ.
  - Proved correctness of the T–satisfiability checker and wrote the space bound $O(k{+}\text{length}(\varphi))$.
  - Specified the bit-encoding macros XOR, ITE, MAJ and used them in the round-contribution formulas $K_{i,t}, S_{i,t}$.
  - Verified the ripple-adder recursion and the lemma linking $\Psi_{\Omega,\Theta}$ to the numeric sum.
  - Wrote the target-equality constraint $\Psi_{\Omega,\Theta}$ and the quantifier prefix $\varphi_{\Omega,\Theta}$.
  - Drew the $D_{\Omega,\Theta}$ diagram (states $D_0 \dots D_5$; registers r_1, r_2, r_3, r_4) and fixed the initial/final configurations.
  - Completed the ($\Rightarrow$) direction and the register-invariant argument for r_1, r_2.
  - Performed final checks.
- **Roll 1009:**
  - Validated the schema examples by enumerating journeys and checking branch totals.
  - Wrote the decision procedure for T–satisfiability and derived the time bound $T_k = O(n^k \,\text{length}(\varphi))$.
  - Implemented the round-contribution terms $K_{i,t}, S_{i,t}$ from bit extracts $\text{bit}_t(\cdot)$ and verified macro expansion.
  - Assembled the ripple-carry adder specification SUM, CARRY and the recursive S_t^j, C_t^j construction.
  - Wrote the pseudocode/emit routine and the worked example section outline.
  - Specified the automaton control flow ("Automaton idea") and state/register roles.
  - Completed the ($\Leftarrow$) direction, the four-lap factorization proof, and the r_3, r_4 flag evolution.
  - Performed final checks.