step$_I$ (C_3, C_4) : asserts that there is a step from the config C_3 to C_4 following the transition $I: (s, b_1, b_2) \to (s', b_2', \varepsilon_1, \varepsilon_2)$.

• For $(\varepsilon_1, \varepsilon_2) = (S, S)$ we have it as:

$$s(C_3) = s \wedge s(C_4) = s' \wedge \exists k', k \; (p'(C_3) = p'(C_4) = k' \wedge p(C_3) = p(C_4) = k \wedge a_k(C_3) = b_2 \wedge a_k(C_4) = b_2' \wedge a_{k'}'(C_3) = b_1 \wedge \langle \text{rest of } \bar{a}, \bar{a}' \text{ unchanged} \rangle \;).$$

• Similarly, for other $\varepsilon_1, \varepsilon_2$.

Remarks about the above formula:

(1) Note that n is fixed but u is free,

(2) '$=$' can be expressed as CNF:

$$a_1 = a_2 \text{ iff } (\bar{a_1} \vee a_2) \wedge (a_1 \vee \bar{a_2}).$$

(3) $(\forall i < T)$ can be expressed as $\wedge$'s.

(4) Writing $\exists k \, (\cdots)$ as CNF is tricky! We

assume that M is an <u>oblivious TM</u>, i.e. the head-position only depends on $|x|$. So, K is known as a fn. of i, and we do not need the quantifier $\exists k$.

(5) $\phi_x(w) \in SAT$ iff $x \in L$.

(6) $|\phi_x(w)| = O(T^2)$.

• This finishes the proof of $L \leq_p SAT$. □

- In fact, a CNF formula ϕ can be reduced to another formula $\psi := \bigwedge_i (v_{i1} \vee v_{i2} \vee v_{i3})$ st. $\psi \in SAT$ iff $\phi \in SAT$.

<u>Proof sketch</u>: Convert a clause with more than 3 literals, eg. $(x_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4)$, to $(x_1 \vee \bar{x}_2 \vee z) \wedge (\bar{z} \vee x_3 \vee \bar{x}_4)$. □

▷ In this way, any clause in k literals can be converted to an equivalent 3-CNF (i.e. conjunction of clauses, each having at most 3 literals).

▷ Thus, 3SAT := {φ | φ is a satisfiable 3CNF formula} is as "hard" as SAT.

Reduction

-Defn: • We call a language A poly-time Karp reducible to a language B if ∃ poly-time TM M s.t.

$$x \in A \text{ iff } M(x) \in B.$$

• We denote this as $A \leq_p B$.

• We say B is NP-hard if

$$\forall A \in NP,\ A \leq_p B.$$

• Further, B is <u>NP-complete</u> if
B is NP-hard & $B \in NP$.

<u>Theorem</u> (Cook, Levin 1971): 3SAT is NP-complete.

<u>Proof</u>:

• We have proved that $\forall L \in NP$,
$L \leq_p$ 3SAT.

• Also, 3SAT $\in NP$. □

– Some easy properties of reductions:

<u>Proposition</u>: (i) $A \leq_p B \leq_p C \Rightarrow A \leq_p C$.

(ii) A is NP-hard & $A \in P \iff P = NP$.

(iii) A is NP-hard & $A \leq_p B \Rightarrow B$ is NP-hard.

– In this sense, the NP-complete problems are the <u>hardest</u> in NP!

– Other egs. of NP-complete problems?

- TSP, Subsum, IntProg, etc. are all NP-complete.

- <u>Several hundreds</u> of NP-complete problems are known!

- We already saw that testing the existence of a boolean feasible point in an integer program is in NP.

▷ IntProg is NP-complete.

<u>Proof:</u>

- Let ϕ be a 3CNF formula in $x_1, \dots, x_n$.
- We will convert each clause in ϕ to a linear inequality in $x_1, \dots, x_n$.
- Eg. convert the clause $(x_1 \vee \bar{x_2} \vee x_3)$ to:

$$\begin{cases} x_1 + (1 - x_2) + x_3 \geq 1 \\ 0 \leq x_1 \leq 1,\ 0 \leq x_2 \leq 1,\ 0 \leq x_3 \leq 1. \end{cases}$$

- Clearly, $3SAT \leq_p$ IntProg. □