# PCPs and Hardness of Approximation

Aditya Morolia

January 30, 2021

# Outline

1. Setting up the definitions
2. PCP Theorems
3. Proof of the PCP Theorem
4. Gap amplification - Proof sketch
5. Alphabet Reduction - Proof sketch

# Setting up the definitions

# Revisiting NP

A language $L$ is said to be in **NP** if there is a poly-time Turing machine $V$ ("verifier") that, given input $x$, checks certificates (or membership proofs) to the effect that $x \in L$. This means,

$$x \in L \implies \exists \pi \text{ s.t. } V^{\pi}(x) = 1$$
$$x \notin L \implies \forall \pi \quad V^{\pi}(x) = 0$$

where $V^{\pi}$ denotes "a verifier with access to certificate $\pi$"

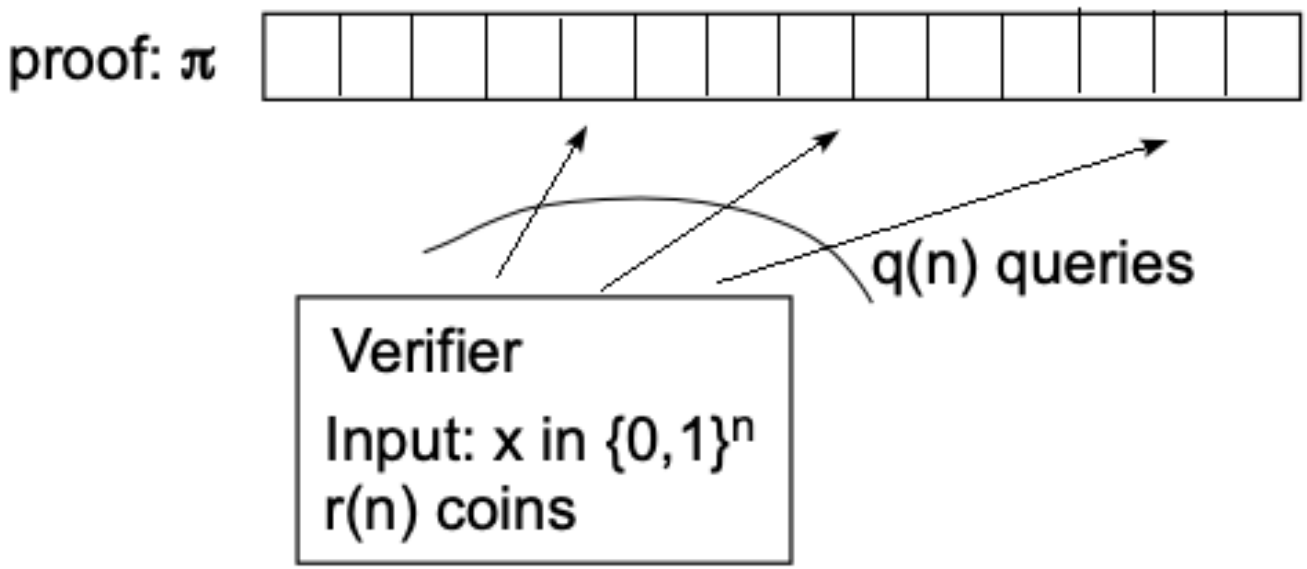
# Probabilistically Checkable Proofs

Consider some changes in previous notion. Let's say the verifier

1. is probabilistic
2. has random access to to the proof string $\pi$.

This becomes possible after allowing queries using a special *address tape* which allows random access. Queries to retrieve a random bit of the proof string.

# Probabilistically Checkable Proofs



# Some Definition

Verifiers can be adaptive and non-adaptive.

## Non-Adaptive Verifier

A nonadaptive verifier selects its queries based only on its input and random tape.

## Adaptive Verifier

An adaptive verifier can in addition rely upon bits it has already queried in $\pi$ to select its next queries.

# $(r, q)$-verifier

## $(r, q)$-verifier

Let $L$ be a language and $q, r : \mathbb{N} \mapsto \mathbb{N}$. We say that $L$ has an $(r(n), q(n))$-verifier if there's a polynomial-time probabilistic algorithm $V$ satisfying:

- **Efficiency:** On input a string $x \in \{0,1\}^n$ and given random access to a string $\pi \in \{0,1\}^*$ (which we call the proof), $V$ uses at most $r(n)$ random coins and makes at most $q(n)$ non-adaptive queries to locations of $\pi$. Then it outputs "1" (for "accept") or "0" (for "reject").

# $(r, q)$-verifier – Continue

## $(r, q)$-verifier

- **Completeness:** If $x \in L$ then there exists a proof $\pi \in \{0,1\}^*$ such that $Pr[V^\pi(x) = 1] = 1$. We call $\pi$ the correct proof for $x$.
- **Soundness:** If $x \notin L$ then for every proof $\pi \in \{0,1\}^*$, $Pr[V^\pi(x) = 1] \leq 1/2$.

# PCPs – an example

The language GNI of pairs of non-isomorphic graphs is in $\mathbf{PCP}(poly(n), 1)$. Say the input for GNI is $\langle G_0, G_1 \rangle$, where $G_0, G_1$ have both $n$ nodes. The verifier expects $\pi$ to contain, for each labeled graph $H$ with $n$ nodes, a bit $\pi[H] \in \{0, 1\}$ corresponding to whether $H \equiv G_0$ or $H \equiv G_1$ ($\pi[H]$ can be arbitrary if neither case holds). In other words, $\pi$ is an (exponentially long) array of bits indexed by the (adjacency matrix representations of) all possible $n$-vertex graphs.

The verifier picks $b \in \{0, 1\}$ at random and a random permutation. She applies the permutation to the vertices of $G_b$ to obtain an isomorphic graph, $H$. She queries the corresponding bit of $\pi$ and accepts iff the bit is $b$.

If $G_0 \not\equiv G_1$, then clearly a proof $\pi$ can be constructed which makes the verifier accept with probability 1. If $G_1 \equiv G_2$, then the probability that any $\pi$ makes the verifier accept is at most $1/2$.

# Constraint satisfaction problem

## CSPs

Let $q, W \in \mathbb{N}$. A $q\text{CSP}_W$ instance $\phi$ is a collection of functions $\phi_1, \ldots \phi_m$ (called constraints) such that each $\phi_i : \{0 \ldots W-1\}^n \to \{0,1\}$ such that each function $\phi_i$ depends on at most $q$ of its input locations. That is, $\forall i \in [m] \exists j_1, \ldots j_q \in [n], f : \{0 \ldots W-1\}^q \to \{0,1\}$ such that $\phi_i(u) = f(u_{j_1}, \ldots u_{j_q}) \forall u \in \{0 \ldots W-1\}^n$

Denote the maximum fraction of constraints satisfied by any assignment $u$ as $val(\phi) = max_u \left( \frac{\sum_1^m \phi_i(u)}{m} \right)$.

# Gap Problems - $\rho$-GAP3SAT

Let $\rho \in \{0, 1\}$. The $\rho$-GAP-3SAT problem is to determine, given a 3CNF formula $\phi$ whether:

- $\phi$ is satisfiable, in which case we say $\phi$ is a YES instance of $\rho$-GAP-3SAT.
- $val(\phi) \leq \rho$, in which case we say $\phi$ is a NO instance of $\rho$-GAP-3SAT.

# Expander Graphs

## Theorem

Let $G = (V, E)$ be a $\lambda$-expander graph for some $\lambda \in (0, 1)$. Let $S$ be a subset of $V$ with $|S| = \beta|V|$ for some $\beta \in (0, 1)$. Let $(X_1, \ldots X_l)$ be a tuple of random variables denoting the vertices of a uniformly chosen $(l-1)$-step path in $G$. Then,

$$(\beta - 2\lambda)^k \leq Pr[\forall_{i \in [l]} X_i \in S] \leq (\beta + 2\lambda)^k$$

## Walsh Hadamard Code

For two strings $x, y \in \{0,1\}^n$, define $x \odot y$ to be the number $\sum_{i=1}^{n} x_i y_i (mod\ 2)$. The Walsh-Hadamard code is the function $WH : \{0,1\}^n \mapsto \{0,1\}^{2^n}$ that maps a string $x \in \{0,1\}^n$ into the string $z \in \{0,1\}^{2^n}$ (a truth table of the function.)

# PCP Theorems

# PCP Theorem

**PCP Theorem**

$\mathsf{NP} = \mathsf{PCP}(\log n, 1)$

$$PCP(r(n), q(n)) \subseteq NTIME(2^{O(r(n))} q(n))$$

**PCP Theorem**

$\exists q \in \mathbb{N}, \rho \in \{0, 1\}$ such that $\rho$-GAP-$q$CSP is NP-hard

# Equivalence of the two definitions

**Theorem 18.2 implies Theorem 18.13.** Assume that $\mathbf{NP} \subseteq \mathbf{PCP}(\log n, 1)$. We will show that $1/2$-GAP $q$CSP is **NP**-hard for some constant $q$. It is enough to reduce a single **NP**-complete language such as 3SAT to $1/2$-GAP $q$CSP for some constant $q$. Under our assumption, 3SAT has a **PCP** system in which the verifier $V$ makes a constant number of queries, which we denote by $q$, and uses $c \log n$ random coins for some constant $c$. Given every input $x$ and $r \in \{0,1\}^{c \log n}$, define $V_{x,r}$ to be the function that on input a proof $\pi$ outputs 1 if the verifier will accept the proof $\pi$ on input $x$ and coins $r$. Note that $V_{x,r}$ depends on at most $q$ locations. Thus for every $x \in \{0,1\}^n$, the collection $\varphi = \{V_{x,r}\}_{r \in \{0,1\}^{c \log n}}$ is a polynomial-sized $q$CSP instance. Furthermore, since $V$ runs in polynomial-time, the transformation of $x$ to $\varphi$ can also be carried out in polynomial-time. By the completeness and soundness of the **PCP** system, if $x \in$ 3SAT then $\varphi$ will satisfy $\mathsf{val}(\varphi) = 1$, while if $x \notin$ 3SAT then $\varphi$ will satisfy $\mathsf{val}(\varphi) \leq 1/2$. ■

# Equivalence of the two definitions

**Theorem 18.13 implies Theorem 18.2.** Suppose that $\rho$-GAP $q$CSP is **NP**-hard for some constants $q$,$\rho < 1$. Then this easily translates into a **PCP** system with $q$ queries, $\rho$ soundness and logarithmic randomness for any language $L$: given an input $x$, the verifier will run the reduction $f(x)$ to obtain a $q$CSP instance $\varphi = \{\varphi_i\}_{i=1}^m$. It will expect the proof $\pi$ to be an assignment to the variables of $\varphi$, which it will verify by choosing a random $i \in [m]$ and checking that $\varphi_i$ is satisfied (by making $q$ queries). Clearly, if $x \in L$ then the verifier will accept with probability 1, while if $x \notin L$ it will accept with probability at most $\rho$. The soundness can be boosted to $1/2$ at the expense of a constant factor in the randomness and number of queries (see Exercise 1). ■

# Equivalence of the two definitions

- PCP verifier (V) $\leftrightarrow$ CSP instance ($\phi$)
- PCP proof ($\pi$) $\leftrightarrow$ Assignment to variables $u$
- Length of proof $\leftrightarrow$ Number of variables (n)
- Number of queries (q) $\leftrightarrow$ Arity of constraints (q)
- Number of random bits (r) $\leftrightarrow$ Logarithm of number of constraints ($\log m$)
- Soundness parameter $\leftrightarrow$ Maximum of val($\phi$) for a NO instance
- NP $\subseteq$ PCP($\log n, 1$) $\leftrightarrow$ $\rho$-GAPqCSP is NP-hard

# Corollary

There exists some constant $\rho < 1$ such that if there is a polynomial-time $\rho$-approximation algorithm for MAX 3SAT then P = NP.

# Gap Producing Reductions

To prove the above corollary for some fixed $\rho < 1$, it suffices to give a polynomial-time reduction $f$ that maps a 3CNF formula to another 3CNF formula such that:

- $\rho \in 3SAT \implies val(f(\phi)) = 1$
- $\rho \notin 3SAT \implies val(f(\phi)) < \rho$

# One Interesting Result

$$NP \subseteq PCP(poly(n), 1)$$

# Proof of the PCP Theorem

## PCP Theorem

$\exists q \in \mathbb{N}, \rho \in \{0,1\}$ such that $\rho$-GAP-$q$CSP is NP-hard

Proof idea: Let $\rho = 1 - \epsilon$. Since the number of satisfied constraints is always a whole number, if $\phi$ is unsatisfiable then $val(\phi) \leq 1 - \frac{1}{m}$. Hence, the gap problem $(1 - \frac{1}{m})$-GAP 3CSP is a generalization of 3SAT and is NP hard. The idea behind the proof is to start with this observation, and iteratively show that $(1 - \epsilon)$-GAP$q$CSP is NP-hard for larger and larger values of $\epsilon$, until $\epsilon$ is as large as some absolute constant independent of $m$. This is formalized in the following lemma.

To formalise the idea from the previous slide, we define a new reduction as follows:

## CL Reduction

A function $f$ mapping CSP instances to a CSP instances is a CL reduction if it is polynomial time computable and for every CSP instance $\phi$ with $m$ constraints and satisfies:

- **Completeness:** $\phi$ is satisfiable $\implies$ $f(\phi)$ is satisfiable
- **Linear Blowup:** $f(\phi)$ has at most $Cm$ constraints and alphabet W, where $C, W$ can depend on the arity and the alphabet size of $\phi$ (but not on the number of constraints or variables).

## PCP Main Lemma

$\exists q_0 \geq 3, \epsilon_0 > 0$ and a CL-reduction $f$ such that for every $q_0$CSP instance $\phi$ on binary alphabet, and every $\epsilon < \epsilon_0$, $\psi = f(\phi)$ is a $q_0$CSP over a binary alphabet such that

$$val(\phi) \leq 1 - \epsilon \implies val(\psi) \leq 1 - 2\epsilon$$

# PCP Theorem from the PCP main lemma

- We can reduce the NP-hard problem $q_0$CSP to $\rho$-GAP-$q_0$CSP
- The idea is to use the CL reduction repeatedly and double the gap in each iteration.
- if $\phi$ is the $q_0$-CSP instance with $m$ constraints, then as observed before, if $\phi$ is satisfiable, $val(\psi) = 1$, otherwise $val(\psi) \leq 1 - \frac{1}{m}$. We can use the gap amplification procedure to amplify this gap (if it exists) from $\frac{1}{m}$ to $val(\psi) \leq \min\{2\epsilon_0, 1 - 2^{\log m}/m\} = 1 - 2\epsilon_0$ by applying this $\log m$ times.
- That is, From the above lemma, the size of $\psi$ will blow up polylogarithmically in $m$. And hence the PCP theorem is proved.

# Gap amplification and Alphabet reduction

## Gap amplification

For every $l \in \mathbb{N}$, there exists a CL-reduction $g_l$ such that for every CSP instance $\phi$ with binary alphabet, the instance $\psi = g_l(\phi)$ has has arity only 2 (but over a non-binary alphabet) and satisfies:

$$\text{val}(\phi) \leq 1 - \epsilon \implies \text{val}(\psi) \leq 1 - l\epsilon$$

for every $\epsilon < \epsilon_0$ where $\epsilon_0 > 0$ is a number depending only on $l$ and the arity $q$ of the original instance $\phi$.

# Gap amplification and Alphabet reduction

## Alphabet reduction

There exists a constant $q_0$ and a CL-reduction $h$ such that for every CSP instance $\phi$, if $\phi$ had arity two over a (possibly non-binary) alphabet $\{0..W-1\}$ then $\psi = h(\phi)$ has arity $q_0$ over a binary alphabet and satisfies:

$$\text{val}(\phi) \leq 1 - \epsilon \implies \text{val}(h(\phi)) \leq 1 - \epsilon/3$$

# Gap amplification and Alphabet reduction

| | Arity | Alphabet | Constraints | Value |
|---|---|---|---|---|
| Original | $q_0$ | binary | $m$ | $1-\epsilon$ |
| | $\Downarrow$ | $\Downarrow$ | $\Downarrow$ | $\Downarrow$ |
| GAP AMPLIFICATION Lemma 18.29 | 2 | $W$ | $Cm$ | $1-6\epsilon$ |
| | $\Downarrow$ | $\Downarrow$ | $\Downarrow$ | $\Downarrow$ |
| ALPHABET REDUCTION Lemma 18.30 | $q_0$ | binary | $C'Cm$ | $1-2\epsilon$ |

# Gap amplification - Proof sketch

# Step 1 - Preparing the constraints for powering

We can convert the given CSP into a CSP with the following properties:

1. The arity $q$ is 2.
2. The constraint graph corresponding to the CSP is $d$-**regular**, where $d$ is a constant independent of alphabet size.
3. The constraint graph is an expander.

This can be done in 2 steps, first adding or splitting constraints to make the graph into a regular graph (we can allow self loops and parallel edges.) Then, overlapping the graph with an expander graph to make a graph with required degree such that it is an expander. The number of constraints will increase, and the gap will reduce because of this. But in the next step, called the powering step, the the gap will increase more than this decrease.

# Step 2 - Powering

## Powering

Let $\psi$ be a $2CSP_W$ instance satisfying Properties 1 through 3. For every number $t$, there is an instance of 2CSP $\psi^t$ such that:

1. $\psi^t$ is a $2CSP_{W'}$-instance with alphabet size $W' < W^{d^{5t}}$, where $d$ denote the degree of $\psi$'s constraint graph. The instance $\psi^t$ has $d^{t+\sqrt{t}}n$ constraints, where $n$ is the number of variables in $\psi$
2. If $\psi$ is satisfiable then so is $\psi^t$.
3. For every $\epsilon < \frac{1}{d\sqrt{t}}$, if $\text{val}(\psi) \leq 1 - \epsilon$ then $\text{val}(\psi^t) \leq 1 - \epsilon'$ for $\epsilon' = \frac{\sqrt{t}}{10^5 dW^4}\epsilon$.
4. The formula $\psi^t$ is computable from $\psi$ in time polynomial in $m$ and $W^{d^t}$.

# Powering - Proof sketch

**Proof:** Let $\psi$ be a $2CSP_W$-instance with $n$ variables and $m = nd$ constraints, and let $G$ denote the constraint graph of $\psi$. The formula $\psi^t$ will have the same number of variables as $\psi$. The new variables $y = y_1, \ldots y_n$ take values over an alphabet of size $W' = W^{d^{5t}}$, and thus a value of a new variable $y_i$ is a $d^{5t}$-tuple of values in $\{0 \ldots W-1\}$. We will think of this tuple as giving a value in $\{0 \ldots W-1\}$ to every old variable $u_j$ where $j$ can be reached from $u_i$ using a path of at most $t + \sqrt{t}$ steps in $G$. In other words, the tuple contains an assignment for every $u_j$ such that $j$ is in the ball of radius $t + \sqrt{t}$ and center $i$ in $G$.

# Powering - Proof sketch

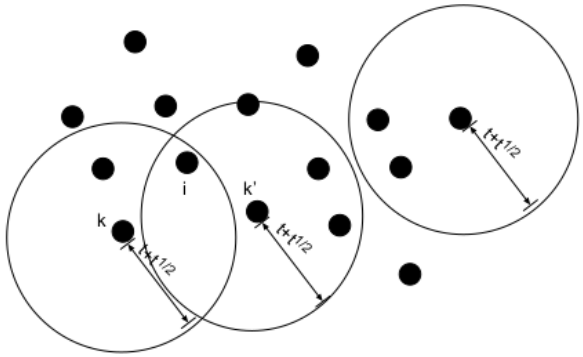



Figure 18.3: An assignment to the formula $\psi^t$ consists of $n$ variables over an alphabet of size less than $W^{d^{5t}}$, where each variable encodes the restriction of an assignment of $\psi$ to the variables that are in some ball of radius $t + \sqrt{t}$ in $\psi$'s constraint graph. Note that an assignment $y_1, \ldots, y_n$ to $\psi^t$ may be *inconsistent* in the sense that if $i$ falls in the intersection of two such balls centered at $k$ and $k'$, then $y_k$ may claim a different value for $u_i$ than the value claimed by $y_{k'}$.

# Powering - Proof sketch

For every $(2t+1)$-step path $p = \langle i_1, \ldots, i_{2t+2} \rangle$ in $G$, we have one corresponding constraint $C_p$ in $\psi^t$ (see Figure 18.4). The constraint $C_p$ depends on the variables $y_{i_1}$ and $y_{i_{2t+1}}$ and outputs FALSE if (and only if) there is some $j \in [2t+1]$ such that:

1. $i_j$ is in the $t + \sqrt{t}$-radius ball around $i_1$.
2. $i_{j+1}$ is in the $t + \sqrt{t}$-radius ball around $i_{2t+2}$
3. If $w$ denotes the value $y_{i_1}$ claims for $u_{i_j}$ and $w'$ denotes the value $y_{i_{2t+2}}$ claims for $u_{i_{j+1}}$, then the pair $(w, w')$ violates the constraint in $\varphi$ that depends on $u_{i_j}$ and $u_{i_{j+1}}$.

# Powering - Proof sketch

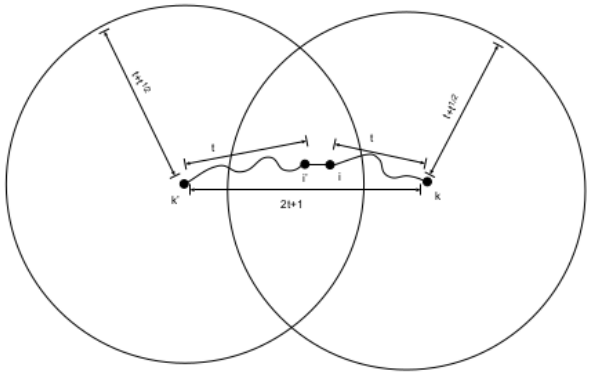



Figure 18.4: $\psi^t$ has one constraint for every path of length $2t+1$ in $\psi$'s constraint graph, checking that the views of the balls centered on the path's two endpoints are consistent with one another and the constraints of $\psi$.

# Alphabet Reduction - Proof sketch

# Alphabet reduction

Alphabet Reduction is a simple consequence of following corollary:

## *qCSP* view of PCP of proximity

There exists positive integer $q_0$ and an exponential-time transformation that given any circuit $C$ of size $m$ and $n$ inputs and two numbers $n_1, n_2$ such that $n_1 + n_2 = n$ produces an instance $\psi_C$ of $q_0 CSP$ of size $2^{poly(m)}$ over a binary alphabet such that:

- The variables can be thought of as being partitioned into three sets $\pi_1, \pi_2, \pi_3$ where $\pi_1$ has $2^{n_1}$ variables and $\pi_2$ has $2^{n_2}$ variables.

- $u_1 \in \{0,1\}^{n_1}$ , $u_2 \in \{0,1\}^{n_2}$ is such that $u_1 \circ u_2$ is a satisfying assignment for circuit $C$ , then there is a string $\pi_3$ of size $2^{poly(m)}$ such that $WH(u_1) \circ WH(u2) \circ \pi_3$ satisfies $\psi_C$.
- For every strings $\pi_1, \pi_2, \pi_3 \in \{0,1\}^*$, where $\pi_1$ and $\pi_2$ have $2n_1$ and $2^{n_2}$ bits respectively, if $\pi_1 \circ \pi_2 \circ \pi_3$ satisfy at least $1/2$ the constraints of $\psi_C$ , then $\pi_1, \pi_2$ are 0.99-close to $WH(u_1), WH(u_2)$ respectively for some $u_1, u_2$ such that $u_1 \circ u_2$ is a satisfying assignment for circuit $C$.

# Alphabet reduction

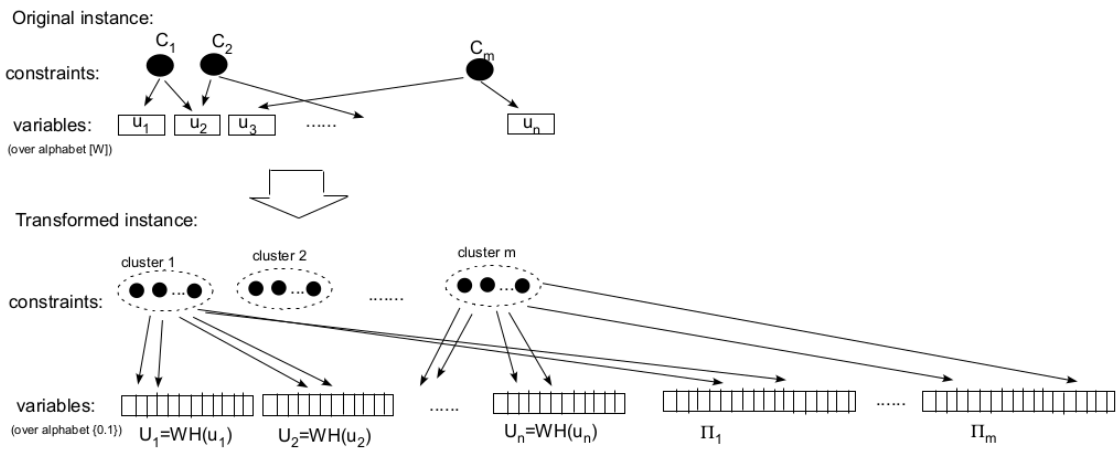



Figure 18.7: The alphabet reduction transformation maps a 2CSP instance $\varphi$ over alphabet $\{0..W-1\}$ into a $q$CSP instance $\psi$ over the binary alphabet. Each variable of $\varphi$ is mapped to a block of binary variables that in the correct assignment will contain the Walsh-Hadamard encoding of this variable. Each constraint $C_\ell$ of $\varphi$ depending on variables $u_i, u_j$ is mapped to a cluster of constraints corresponding to all the **PCP** of proximity constraints for $C_\ell$. These constraint depend on the encoding of $u_i$ and $u_j$, and on additional auxiliary variables that in the correct assignment contain the **PCP** of proximity proof that these are indeed encoding of values that make the constraint $C_\ell$ true.

Thank You!