online multiplication

1 Online computation

idea:

- input sequence $(I_0, I_1, \ldots)$
- output sequence $(O_0, O_1, \ldots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

1 Online computation

idea:

- input sequence $(I_0, I_1, \ldots)$
- output sequence $(O_0, O_1, \ldots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

Online k-tape Turing machines: get extra

- read only input tape
- write only output tape
- other tapes with numbers $\tau \in [1:k]$: work tapes
- $t(n)$-time bounded if for all inputs output O_n is produced after at most $t(n)$ steps for all $n \geq n_0$.

1 Online computation

idea:

- input sequence $(I_0, I_1, \dots)$
- output sequence $(O_0, O_1, \dots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

Online k-tape Turing machines: get extra

- read only input tape
- write only output tape
- other tapes with numbers $\tau \in [1:k]$: work tapes
- $t(n)$-time bounded if for all inputs output O_n is produced after at most $t(n)$ steps for all $n \geq n_0$.

binary multiplication:

$$\cdot_n : \mathbb{B}^n \times \mathbb{B}^n \to B^{2n}$$

with

$$a \cdot_n b = bin_{2n}(\langle a \rangle \cdot \langle b \rangle)$$

1 Online computation

idea:

- input sequence $(I_0, I_1, \dots)$
- output sequence $(O_0, O_1, \dots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

Online k-tape Turing machines: get extra

- read only input tape
- write only output tape
- other tapes with numbers $\tau \in [1:k]$: work tapes
- $t(n)$-time bounded if for all inputs output O_n is produced after at most $t(n)$ steps for all $n \geq n_0$.

binary multiplication:

$$\cdot_n : \mathbb{B}^n \times \mathbb{B}^n \to B^{2n}$$

with

$$a \cdot_n b = bin_{2n}(\langle a \rangle \cdot \langle b \rangle)$$

online multiplication: with operands

$$X[N-1:0]\ ,\ K[N-1:0] \in \mathbb{B}^N$$

- inputs: $I_i = (X_i, K_i)$ on 2 tracks
- outputs: $O_i = (X[i-1:0] \cdot_{2i} K[i-1:0])_i$

1 Online computation

idea:

- input sequence $(I_0, I_1, \ldots)$
- output sequence $(O_0, O_1, \ldots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

Online k-tape Turing machines: get extra

- read only input tape
- write only output tape
- other tapes with numbers $\tau \in [1:k]$: work tapes
- $t(n)$-time bounded if for all inputs output O_n is produced after at most $t(n)$ steps for all $n \geq n_0$.

binary multiplication:

$$\cdot_n : \mathbb{B}^n \times \mathbb{B}^n \to B^{2n}$$

with

$$a \cdot_n b = bin_{2n}(\langle a \rangle \cdot \langle b \rangle)$$

online multiplication: with operands

$$X[N-1:0]\ ,\ K[N-1:0] \in \mathbb{B}^N$$

- inputs: $I_i = (X_i, K_i)$ on 2 tracks
- outputs: $O_i = (X[i-1:0] \cdot_{2i} K[i-1:0])_i$

Why don't we have to revise previous outputs? :

Lemma 1. *The i low order bits of $A \cdot_n B$ depend only of the i low order bist of A and B*

1 Online computation

idea:

- input sequence $(I_0, I_1, \ldots)$
- output sequence $(O_0, O_1, \ldots)$
- for all i you have to produce output O_i before reading input I_{i+1}

This extra structure of the computation makes proving lower bounds *much* easier.

Online k-tape Turing machines: get extra

- read only input tape
- write only output tape
- other tapes with numbers $\tau \in [1:k]$: work tapes
- $t(n)$-time bounded if for all inputs output O_n is produced after at most $t(n)$ steps for all $n \geq n_0$.

binary multiplication:

$$\cdot_n : \mathbb{B}^n \times \mathbb{B}^n \to B^{2n}$$

with

$$a \cdot_n b = bin_{2n}(\langle a \rangle \cdot \langle b \rangle)$$

online multiplication: with operands

$$X[N-1:0]\ ,\ K[N-1:0] \in \mathbb{B}^N$$

- inputs: $I_i = (X_i, K_i)$ on 2 tracks
- outputs: $O_i = (X[i-1:0] \cdot_{2i} K[i-1:0])_i$

Why don't we have to revise previous outputs? :

Lemma 1. *The i low order bits of $A \cdot_n B$ depend only of the i low order bist of A and B*

- decompose $X \in \mathbb{B}^n$ as

$$\begin{aligned} X_H &= X[n-1:i] \\ X_L &= X[i-1:0] \end{aligned}$$

Then

$$\langle x \rangle = \langle x_H \rangle \cdot 2^i + \langle x_L \rangle$$

-

$$\begin{aligned} \langle A \rangle \cdot \langle B \rangle &= \langle A_H \rangle \cdot \langle B_H \rangle \cdot 2^{2i} \\ &+ (\langle A_H \rangle \cdot \langle B_L \rangle + \langle A_L \rangle \cdot \langle B_H \rangle) \cdot 2^i \\ &+ \langle A_L \rangle \cdot \langle B_L \rangle \end{aligned}$$

2 Stating the lower bound

Ω-notation: 2 Variants for $f(n) = \Omega(g(n))$

- for almost all n:

$$\exists c > 0,\ n_0\ \forall n \geq n_0. \quad f(n) \geq c \cdot g(n)$$

- *here* infinitely often:

$$\exists c > 0\ \forall m\ \exists n \geq m. \quad f(n) \geq c \cdot g(n)$$

2 Stating the lower bound

Ω-notation: 2 Variants for $f(n) = \Omega(g(n))$

- for almost all n:

$$\exists c > 0,\ n_0\ \forall n \geq n_0.\quad f(n) \geq c \cdot g(n)$$

- *here* infinitely often:

$$\exists c > 0\ \forall m\ \exists n \geq m.\quad f(n) \geq c \cdot g(n)$$

Lemma 2. *Online multiplication by k-tape Turing machines requires time* $\Omega(n \log n)$.

2 Stating the lower bound

Ω-notation: 2 Variants for $f(n) = \Omega(g(n))$

- for almost all n:

$$\exists c > 0,\ n_0\ \forall n \geq n_0. \quad f(n) \geq c \cdot g(n)$$

- *here* infinitely often:

$$\exists c > 0\ \forall m\ \exists n \geq m. \quad f(n) \geq c \cdot g(n)$$

Lemma 2. *Online multiplication by k-tape Turing machines requires time* $\Omega(n \log n)$.

- 1969: S. Cook (NP-completeness) and S.O. Aanderaa (his advisor), 23 pages
- 1974: M.S. Paterson (advisor of Valiant, 233 descendants), M.J. Fisher (parallel prefix, Presburger arithmetic), A.R. Meyer (MIT). 15 pages
- 1982: S. Reisch, G. Schnitger (my students): 2 pages + another 2 pages from Paterson et al.

2 Stating the lower bound

Ω-notation: 2 Variants for $f(n) = \Omega(g(n))$

- for almost all n:

$$\exists c > 0,\ n_0\ \forall n \geq n_0.\quad f(n) \geq c \cdot g(n)$$

- *here* infinitely often:

$$\exists c > 0\ \forall m\ \exists n \geq m.\quad f(n) \geq c \cdot g(n)$$

Lemma 2. *Online multiplication by k-tape Turing machines requires time* $\Omega(n \log n)$.

- 1969: S. Cook (NP-completeness) and S.O. Aanderaa (his advisor), 23 pages
- 1974: M.S. Paterson (advisor of Valiant, 233 descendants), M.J. Fisher (parallel prefix, Presburger arithmetic), A.R. Meyer (MIT). 15 pages
- 1982: S. Reisch, G. Schnitger (my students): 2 pages + another 2 pages from Paterson et al.

proof ideas:

- overlap argument using edge partition in the style of 'determinism versus nondeterminism'
- multiplication is not hard for all operands. E.g. multiplication with 0 is trivial. Multiplication with 2^n is a shift.
- special sequence of first operands

$$K_N \in \mathbb{B}^N \quad , \quad K_N[i] = 1 \leftrightarrow \exists r \in \mathbb{N}_0.\ i = 2^r$$

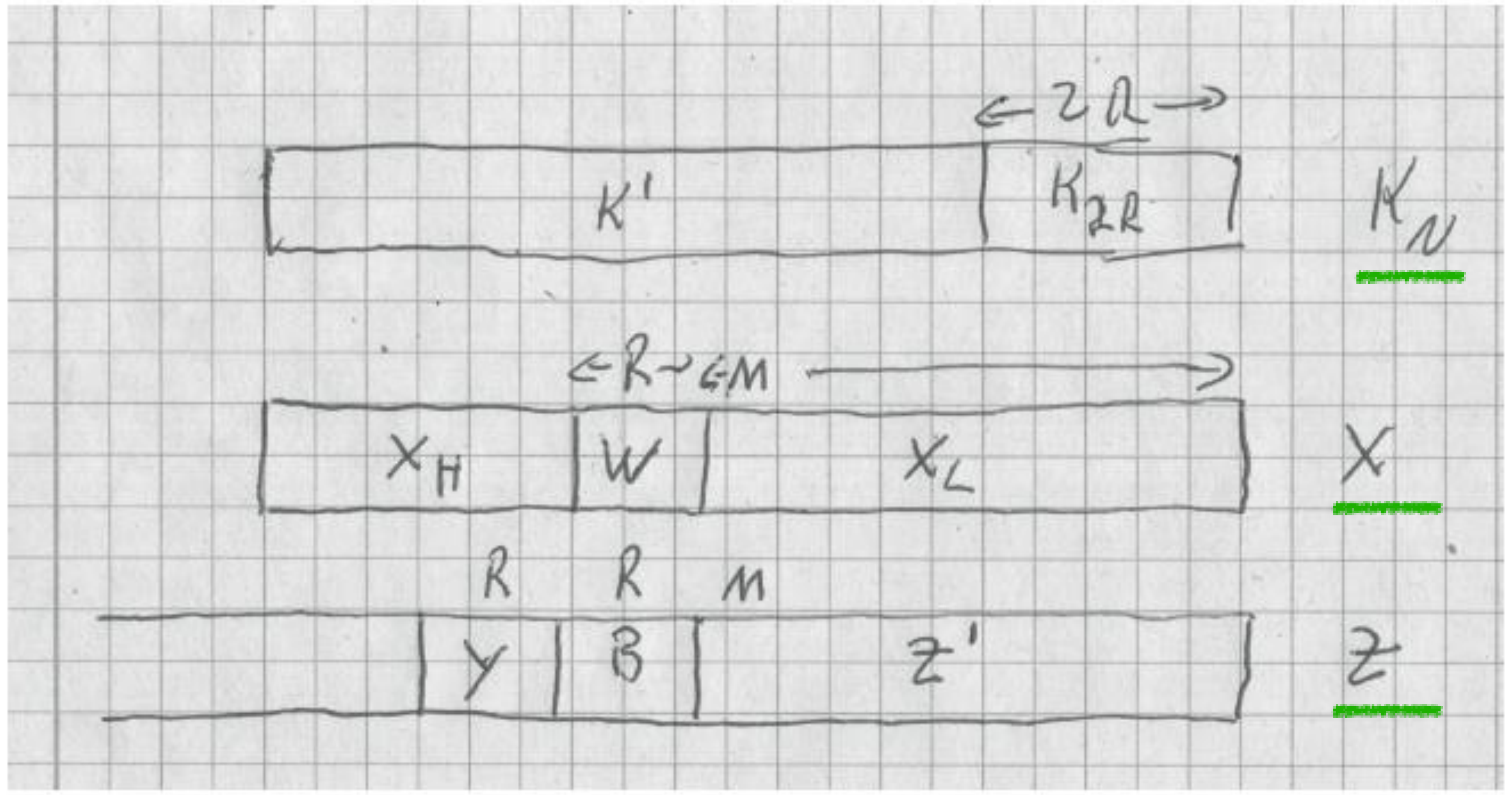
  Ones at positions which are powers of two.

$$K_N = \ldots 10000000100010110$$

$$k_N = \langle K_N \rangle = \sum_{2^r < N} 2^{2^r}$$

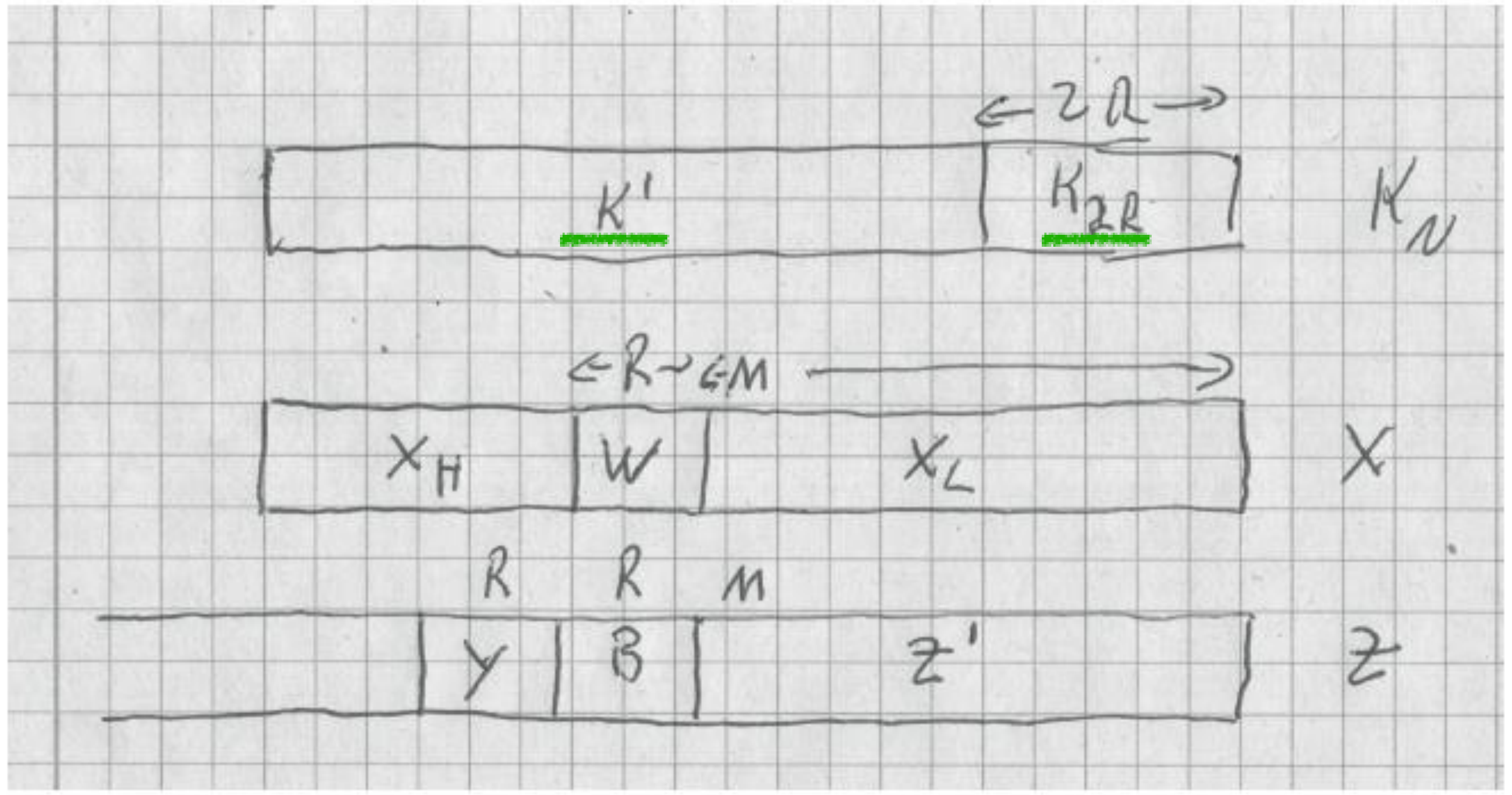
- 1969, 1974: argue for operands of length N about average time taken over all second operands X.
- 1982: for each N argue with *one* Kolmogorov-random operand X: short computations would allow to compress the X operand.

3 Reconstructing operand bits from result bits



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular

3 Reconstructing operand bits from result bits

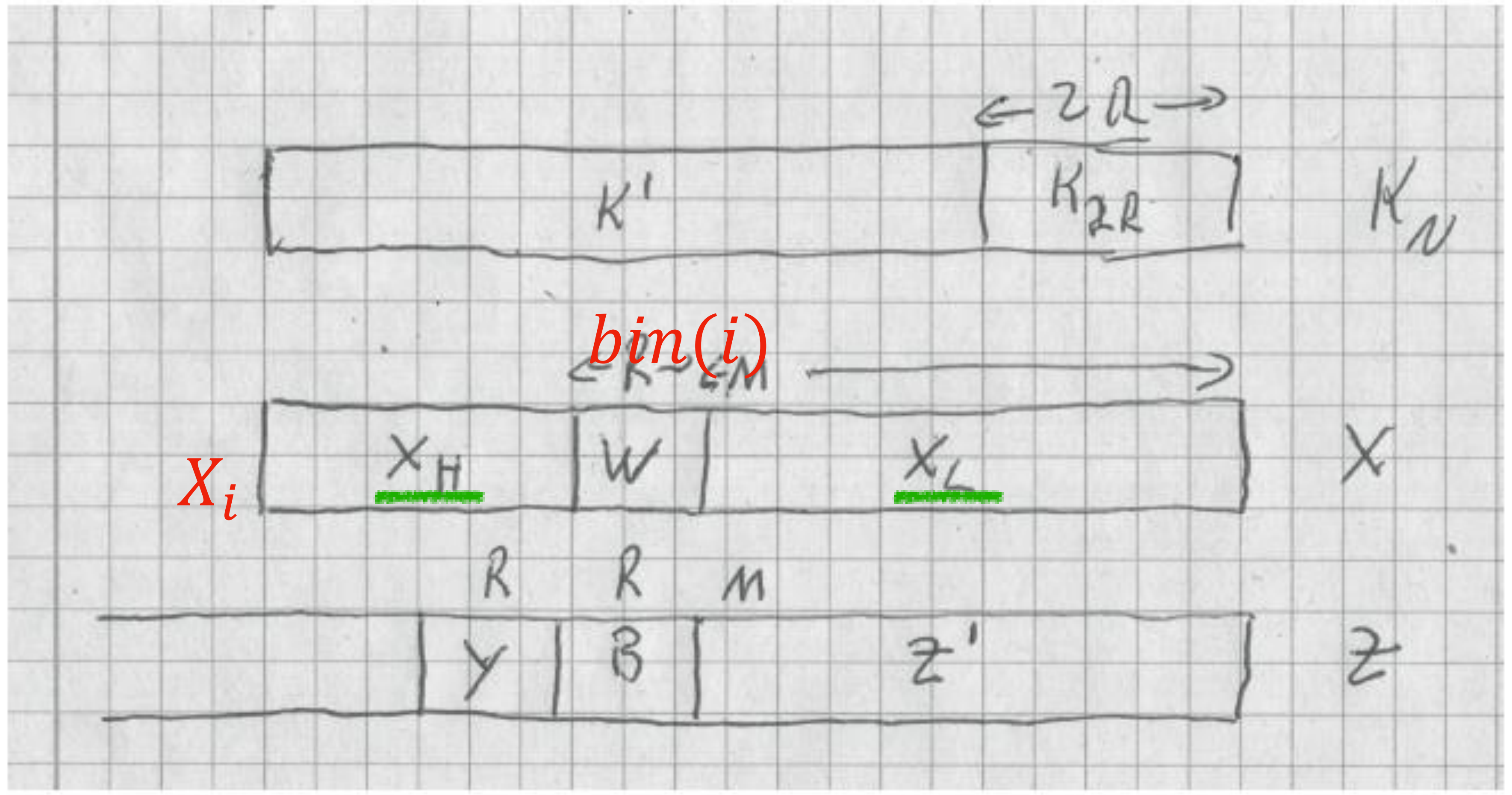


- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular

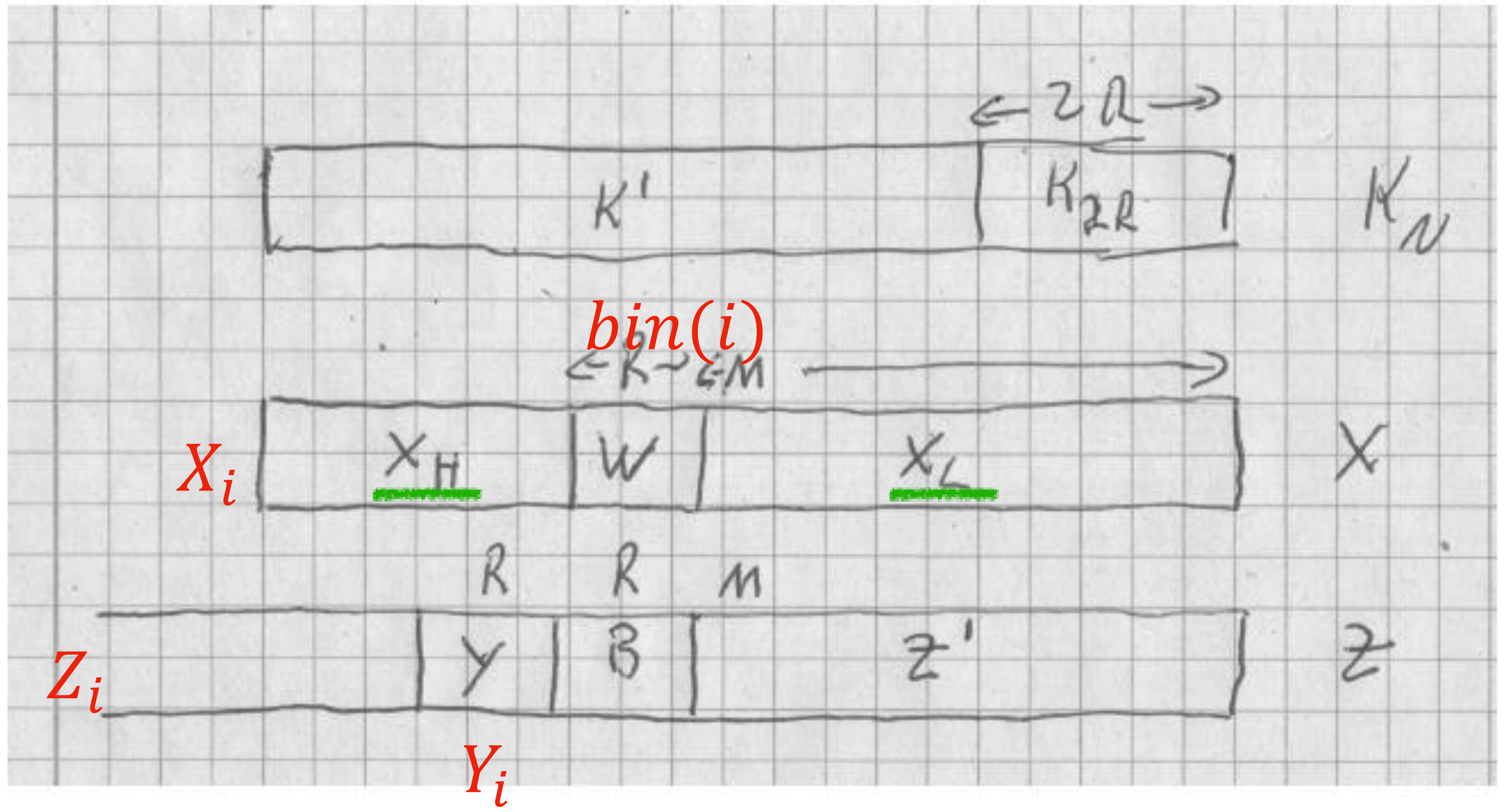
-

$$K_N = K' \circ K_{2R}$$

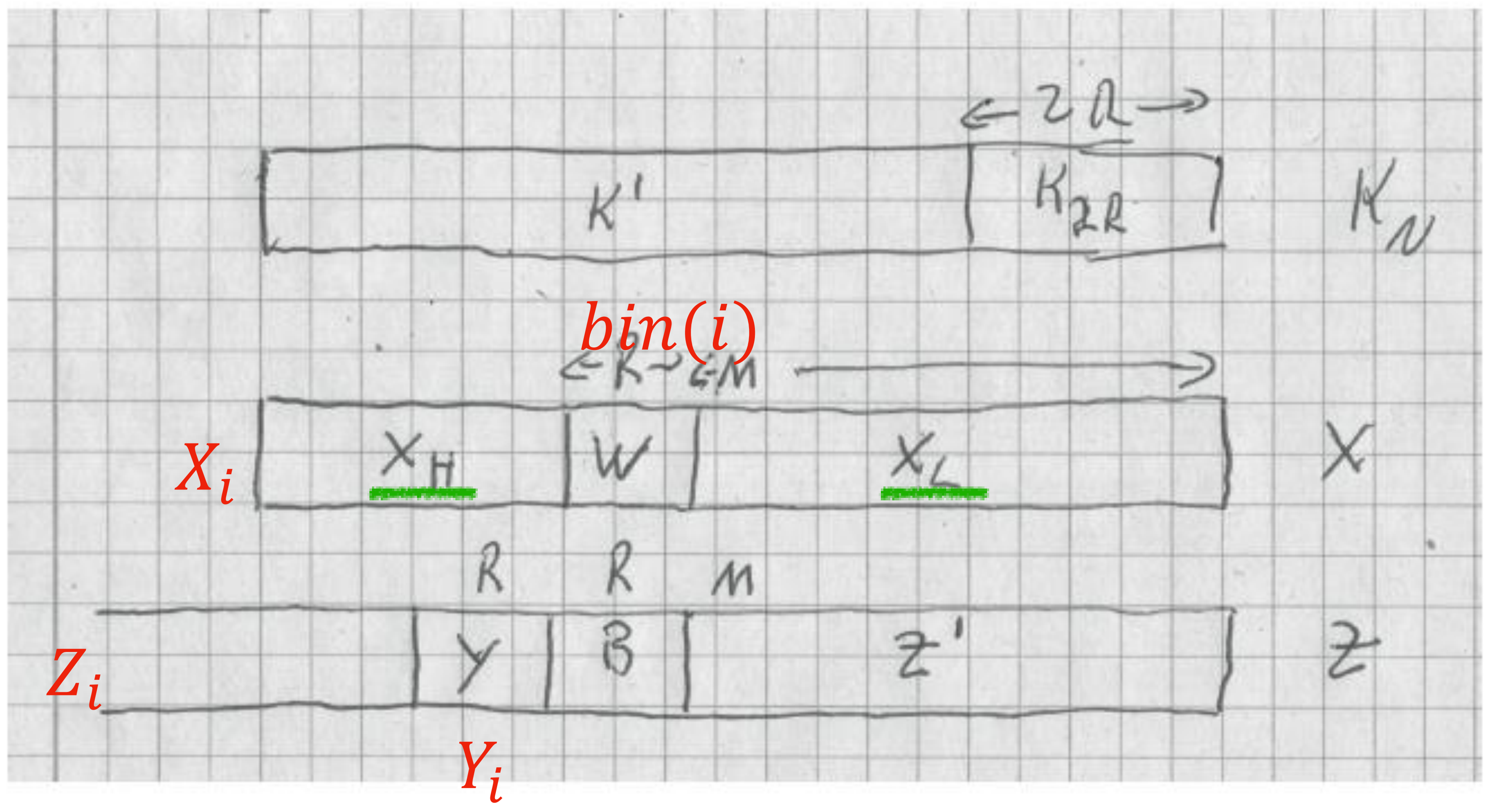
3 Reconstructing operand bits from result bits



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular
-
$$K_N = K' \circ K_{2R}$$
- W block of X starting at position M
$$W = X[M+R-1 : M]$$
- Fix portions X_L, X_H of X outside of W. For $i \in [0 : 2^R - 1]$ set W to $bin_R(i)$ to obtain
$$X_i = X_H \circ bin_R(i) \circ X_L$$



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular
- $$K_N = K' \circ K_{2R}$$
- W block of X starting at position M
$$W = X[M + R - 1 : M]$$
- Fix portions X_L, X_H of X outside of W. For $i \in [0 : 2^R - 1]$ set W to $bin_R(i)$ to obtain
$$X_i = X_H \circ bin_R(i) \circ X_L$$
- multiplication of K_N with X_i gives result
$$Z_i = K_N \cdot_N X_i$$
and for block Y
$$Y_i = Z[M + 2R - 1 : M + R]$$



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular

- $$K_N = K' \circ K_{2R}$$

- W block of X starting at position M

$$W = X[M+R-1 : M]$$

- Fix portions X_L, X_H of X outside of W. For $i \in [0 : 2^R - 1]$ set W to $bin_R(i)$ to obtain

$$X_i = X_H \circ bin_R(i) \circ X_L$$

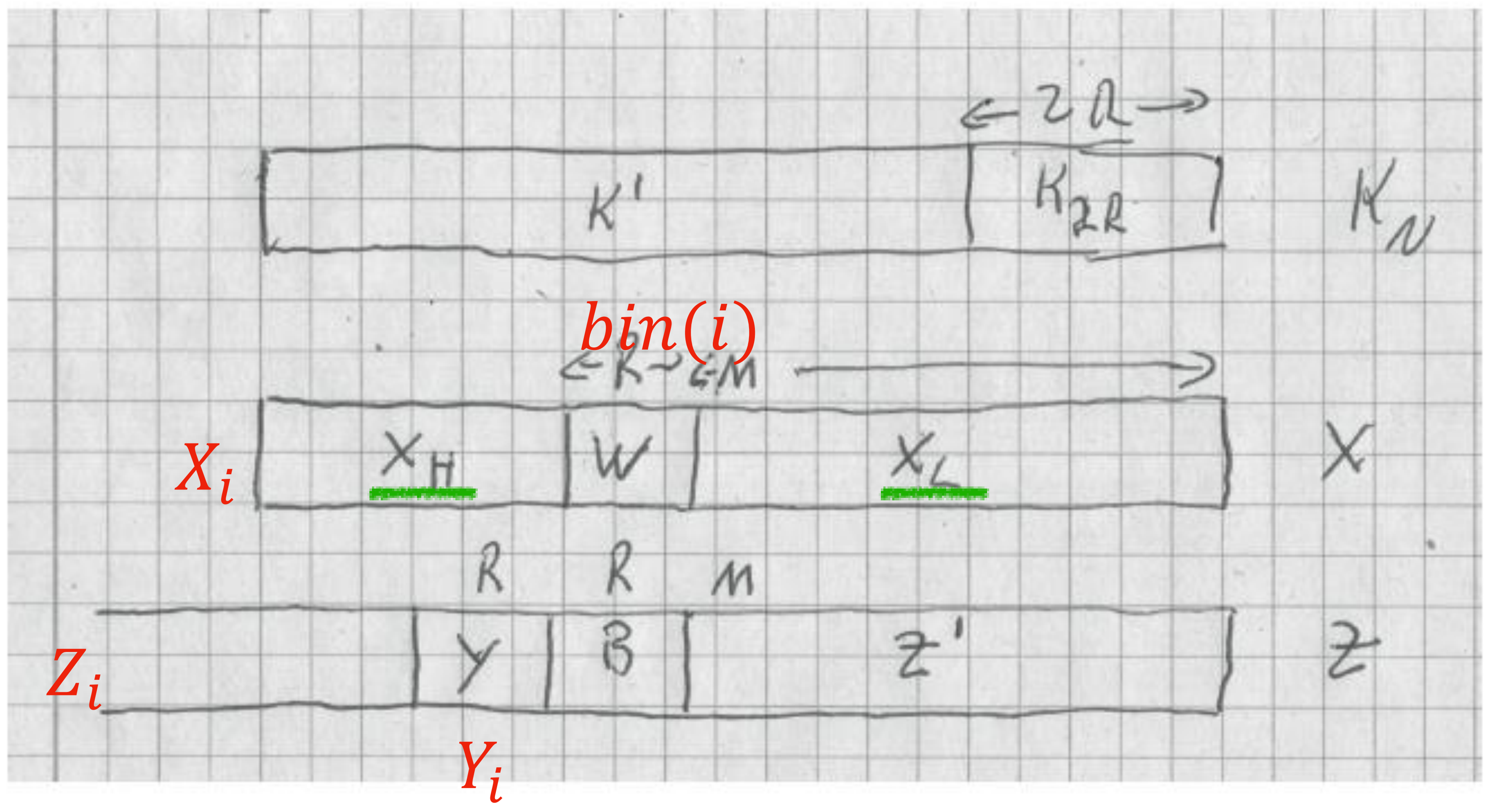
- multiplication of K_N with X_i gives result

$$Z_i = K_N \cdot_N X_i$$

and for block Y

$$Y_i = Z[M+2R-1 : M+R]$$

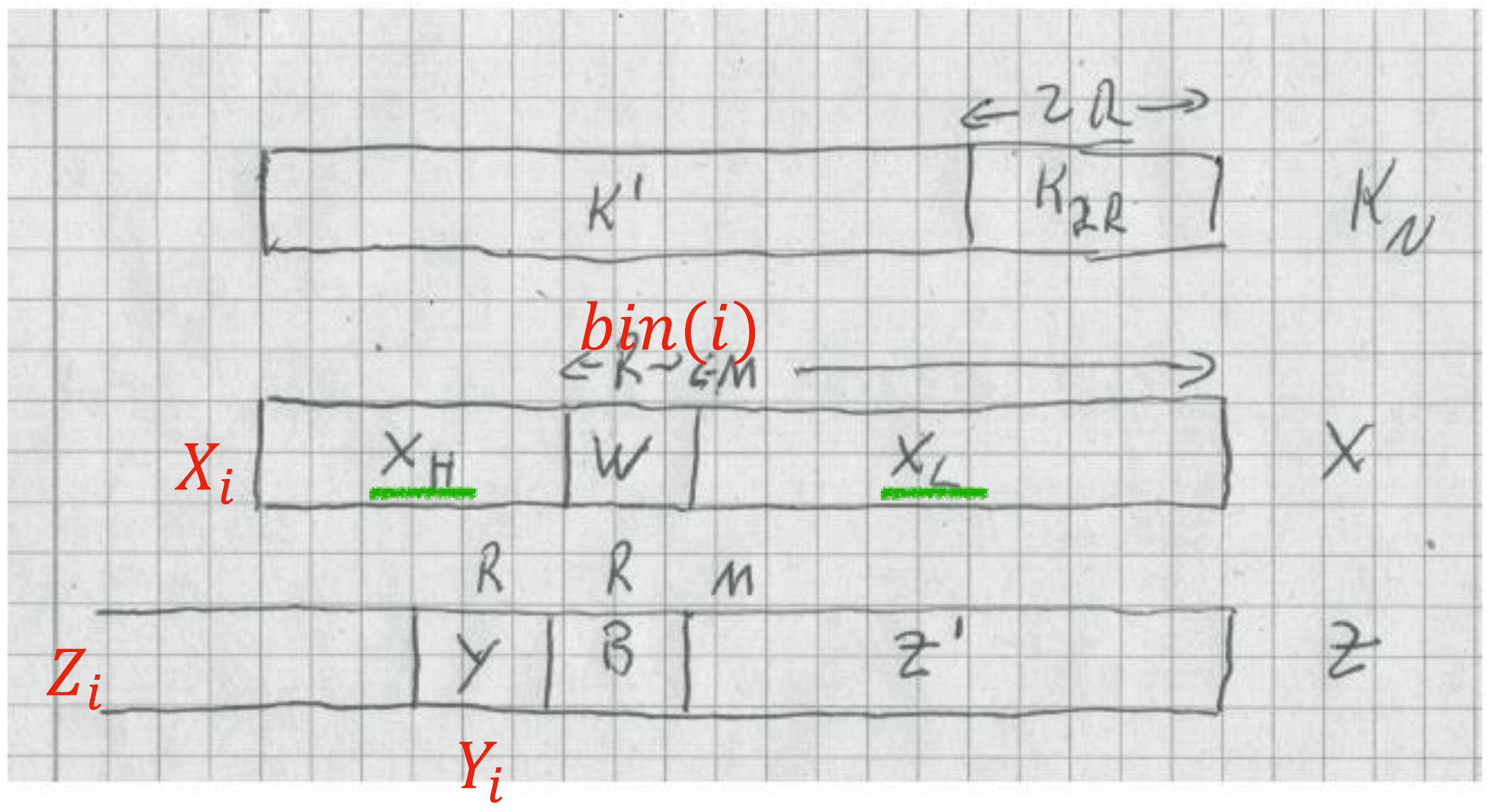
Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular
-
$$K_N = K' \circ K_{2R}$$
- W block of X starting at position M
$$W = X[M+R-1:M]$$
- Fix portions X_L, X_H of X outside of W. For $i \in [0:2^R-1]$ set W to $bin_R(i)$ to obtain
$$X_i = X_H \circ bin_R(i) \circ X_L$$
- multiplication of K_N with X_i gives result
$$Z_i = K_N \cdot_N X_i$$
and for block Y
$$Y_i = Z[M+2R-1:M+R]$$

Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i
$$x_i = \langle X_i \rangle \text{ , } y_i = \langle Y_i \rangle \text{ , } z_i = \langle Z_i \rangle \text{ ,} k_i = \langle K_i \rangle \text{ ,} k' = \langle K' \rangle \text{ , } k_{2R} = \langle K_{2R} \rangle$$
for $i < j$:
$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j-i) \cdot 2^M \cdot k_N \end{aligned}$$



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular

-
$$K_N = K' \circ K_{2R}$$

- W block of X starting at position M
$$W = X[M+R-1:M]$$

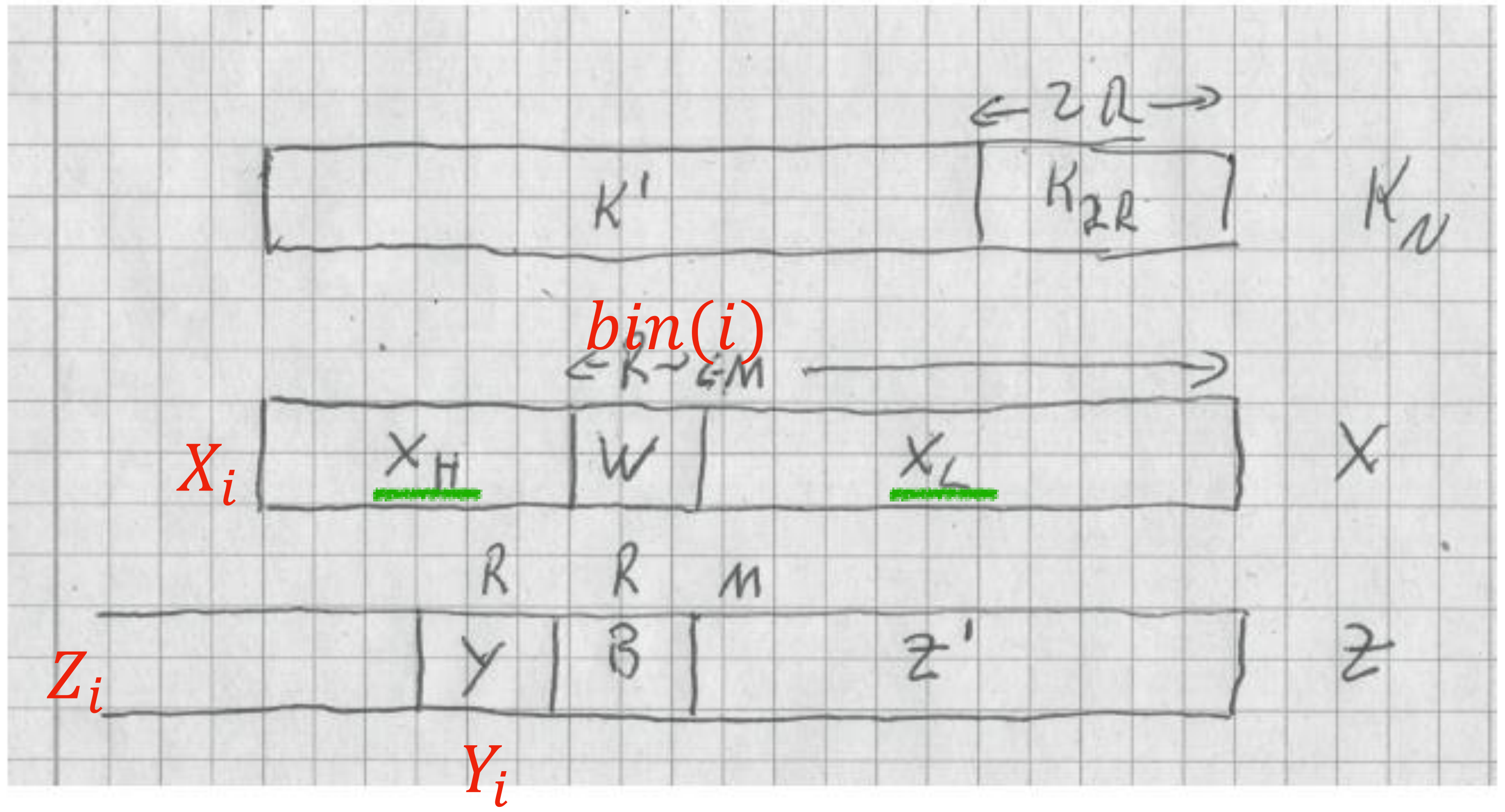
- Fix portions X_L, X_H of X outside of W. For $i \in [0:2^R-1]$ set W to $bin_R(i)$ to obtain
$$X_i = X_H \circ bin_R(i) \circ X_L$$

- multiplication of K_N with X_i gives result
$$Z_i = K_N \cdot_N X_i$$
and for block Y
$$Y_i = Z[M+2R-1:M+R]$$

Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i
$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$
for $i < j$:
$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j-i) \cdot 2^M \cdot k_N \end{aligned}$$

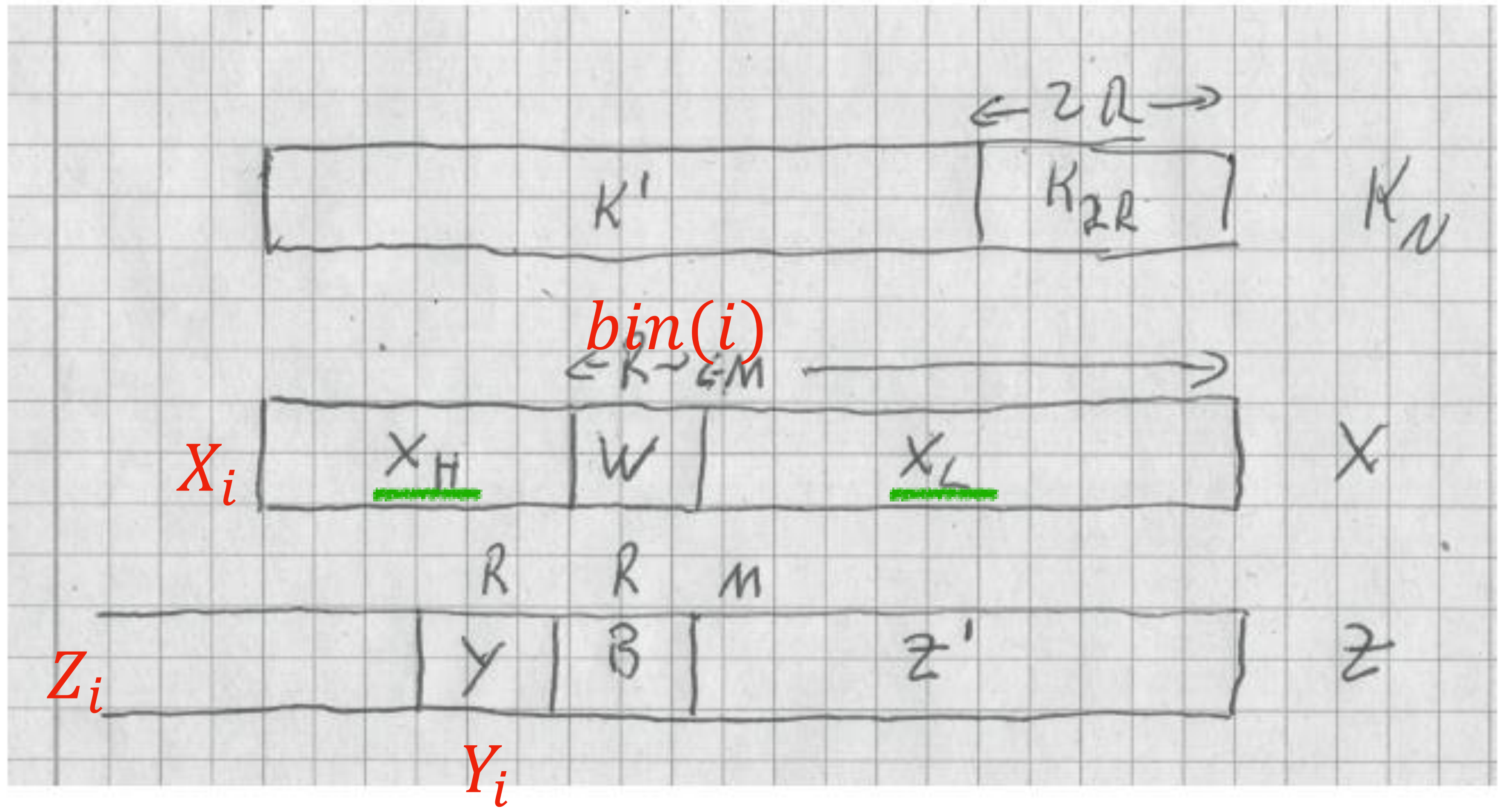
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$
$$z_j - z_i \equiv (j-i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$



- decompose N bit operands X and K_N and result $Z = K_N \cdot_N X$ as shown in figure 1 in blocks of length $R = 2^r$. In particular
-
$$K_N = K' \circ K_{2R}$$
- W block of X starting at position M
$$W = X[M + R - 1 : M]$$
- Fix portions X_L, X_H of X outside of W. For $i \in [0 : 2^R - 1]$ set W to $bin_R(i)$ to obtain
$$X_i = X_H \circ bin_R(i) \circ X_L$$
- multiplication of K_N with X_i gives result
$$Z_i = K_N \cdot_N X_i$$
and for block Y
$$Y_i = Z[M + 2R - 1 : M + R]$$

Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i
$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$
for $i < j$:
$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j - i) \cdot 2^M \cdot k_N \end{aligned}$$
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$
$$z_j - z_i \equiv (j - i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$
- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal
$$Z_i[M - 1 : 0] = Z_j[M - 1 : 0] = Z'$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

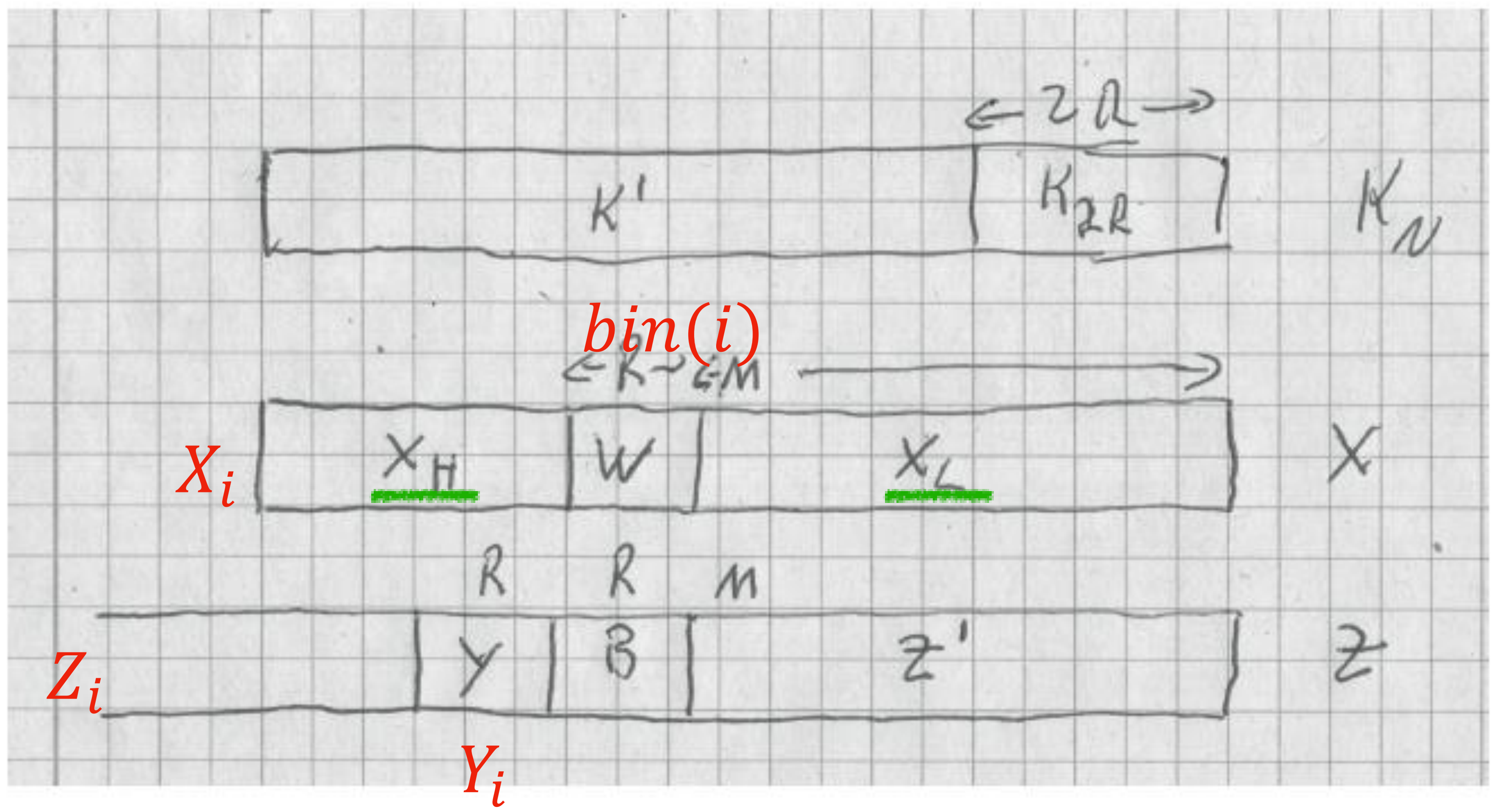
$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j - i) \cdot 2^M \cdot k_N \end{aligned}$$

- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j - i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

  for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j - i) \cdot 2^M \cdot k_N \end{aligned}$$

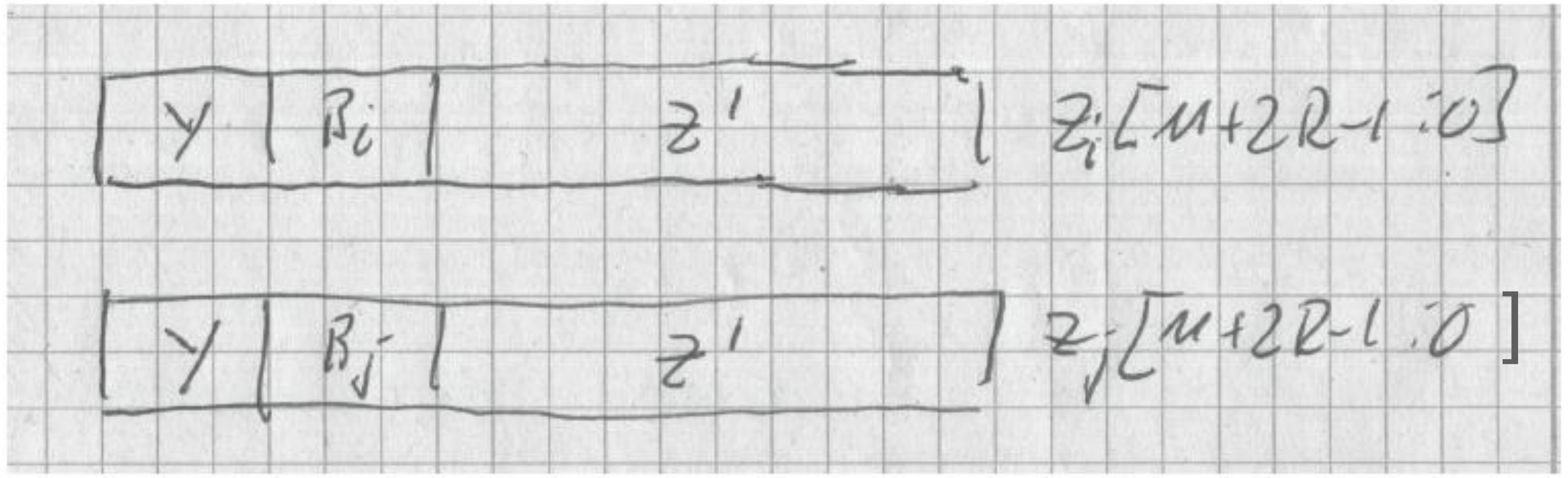
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j - i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

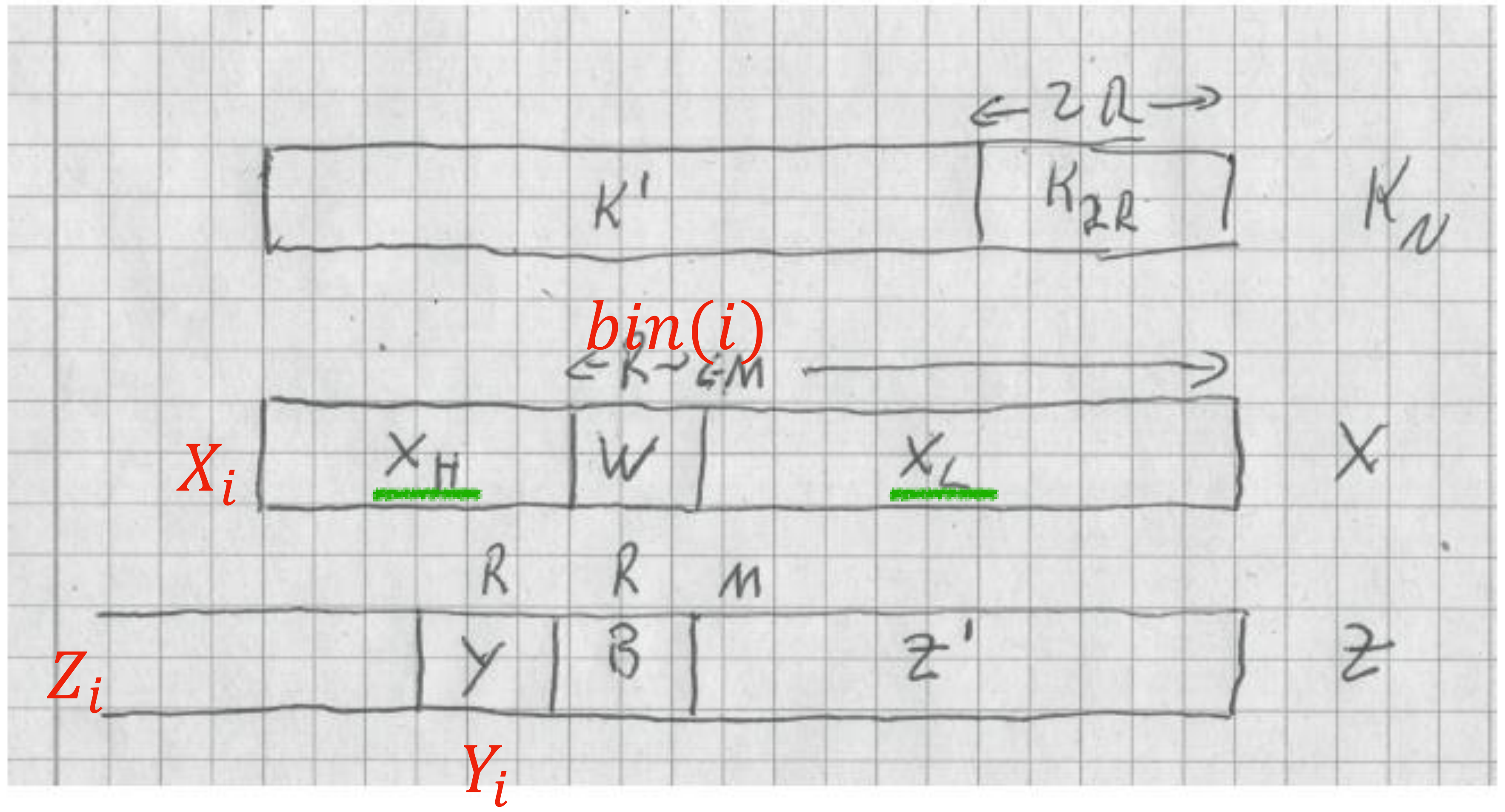
- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \end{aligned}$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j - i) \cdot 2^M \cdot k_N \end{aligned}$$

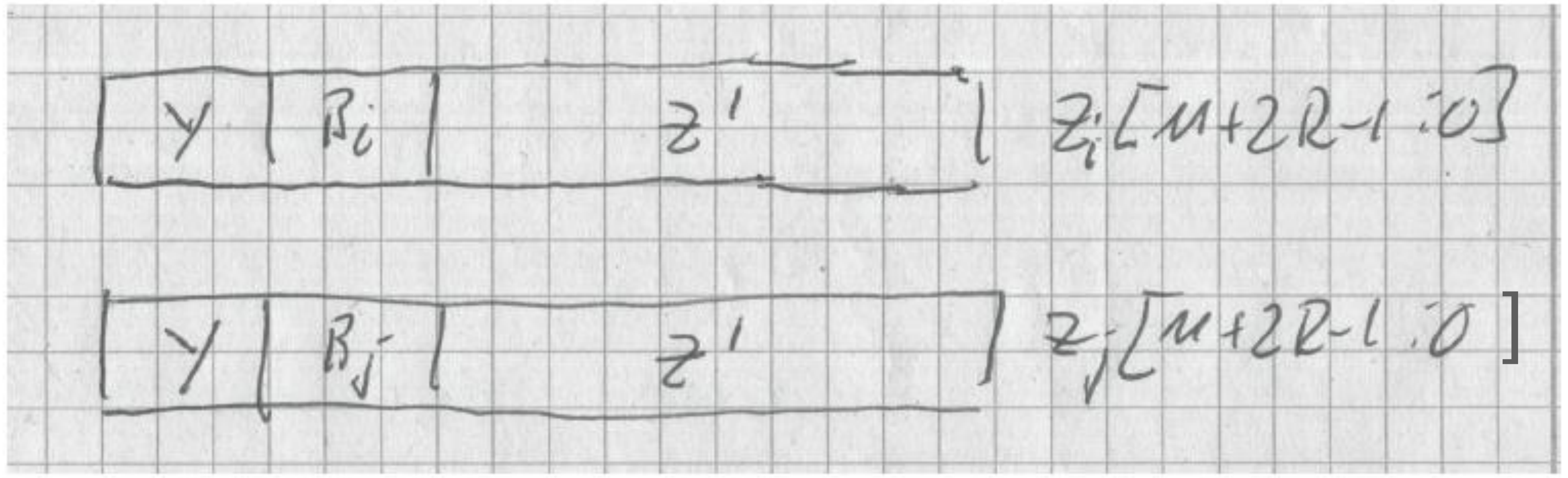
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j - i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

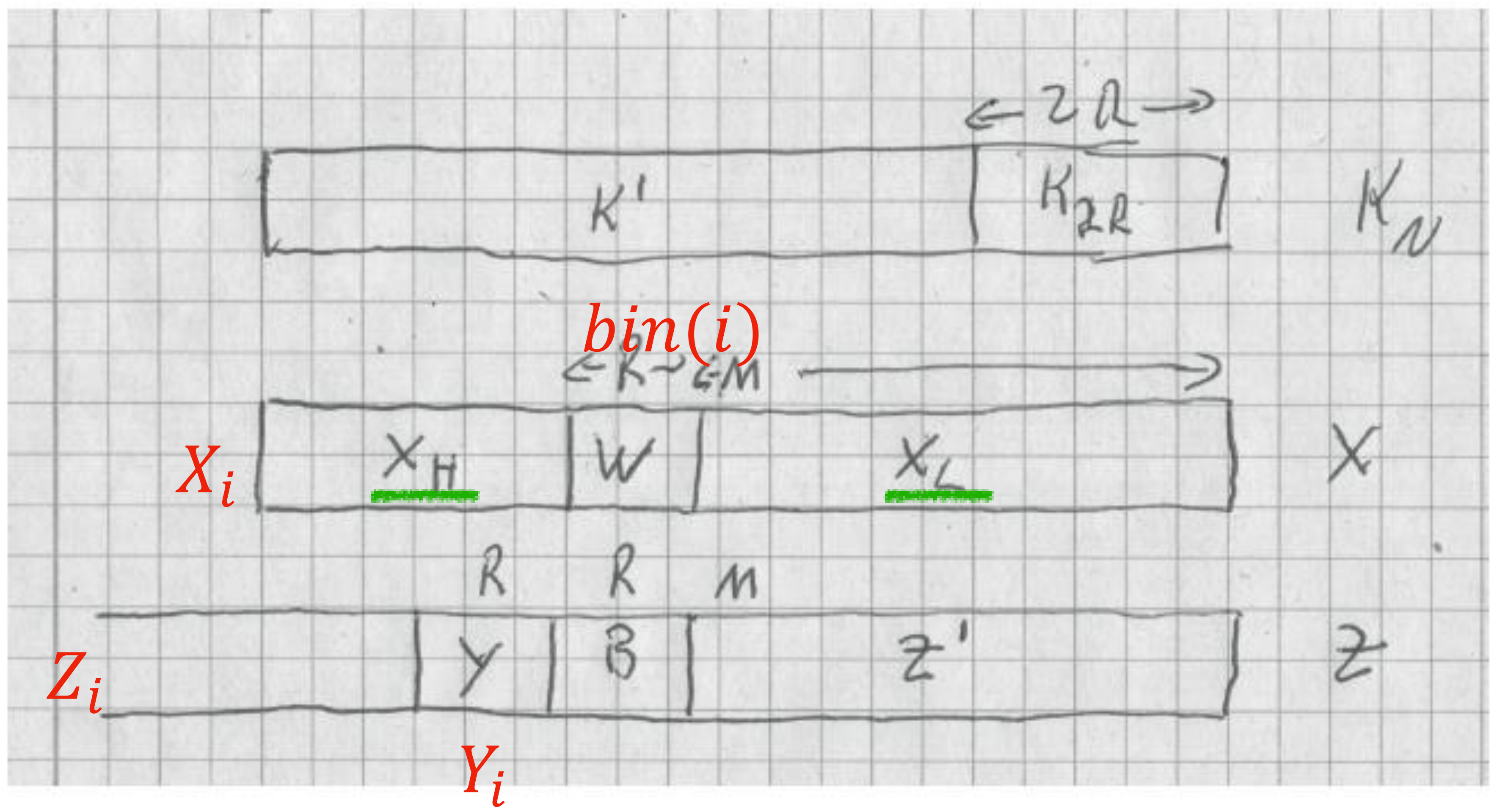
$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \\ z_j - z_i &\equiv (\langle B_j \rangle - \langle B_i \rangle) \cdot 2^M \pmod{2^{M+2R}} \\ (\langle B_j \rangle - \langle B_i \rangle) &= a \in \mathbb{Z} \quad , \quad |a| < 2^R \\ \end{aligned}$$

$$\boxed{(j - i) \cdot k_{2R} \equiv a \pmod{2^{2R}}}$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j - i) \cdot 2^M \cdot k_N \end{aligned}$$

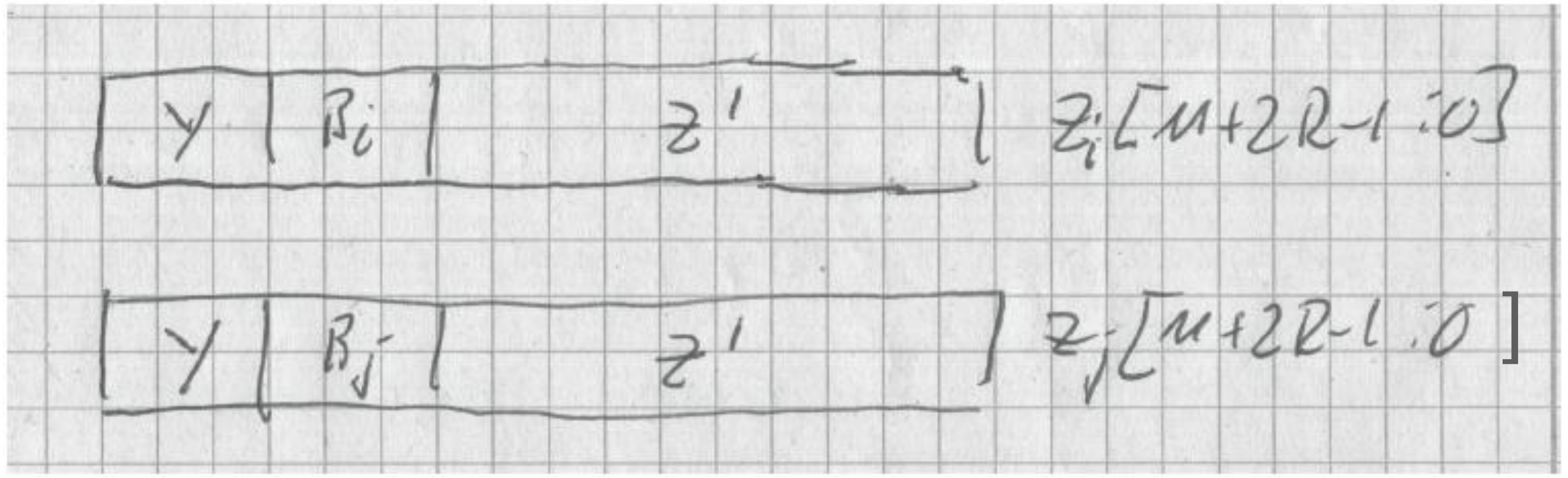
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j - i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



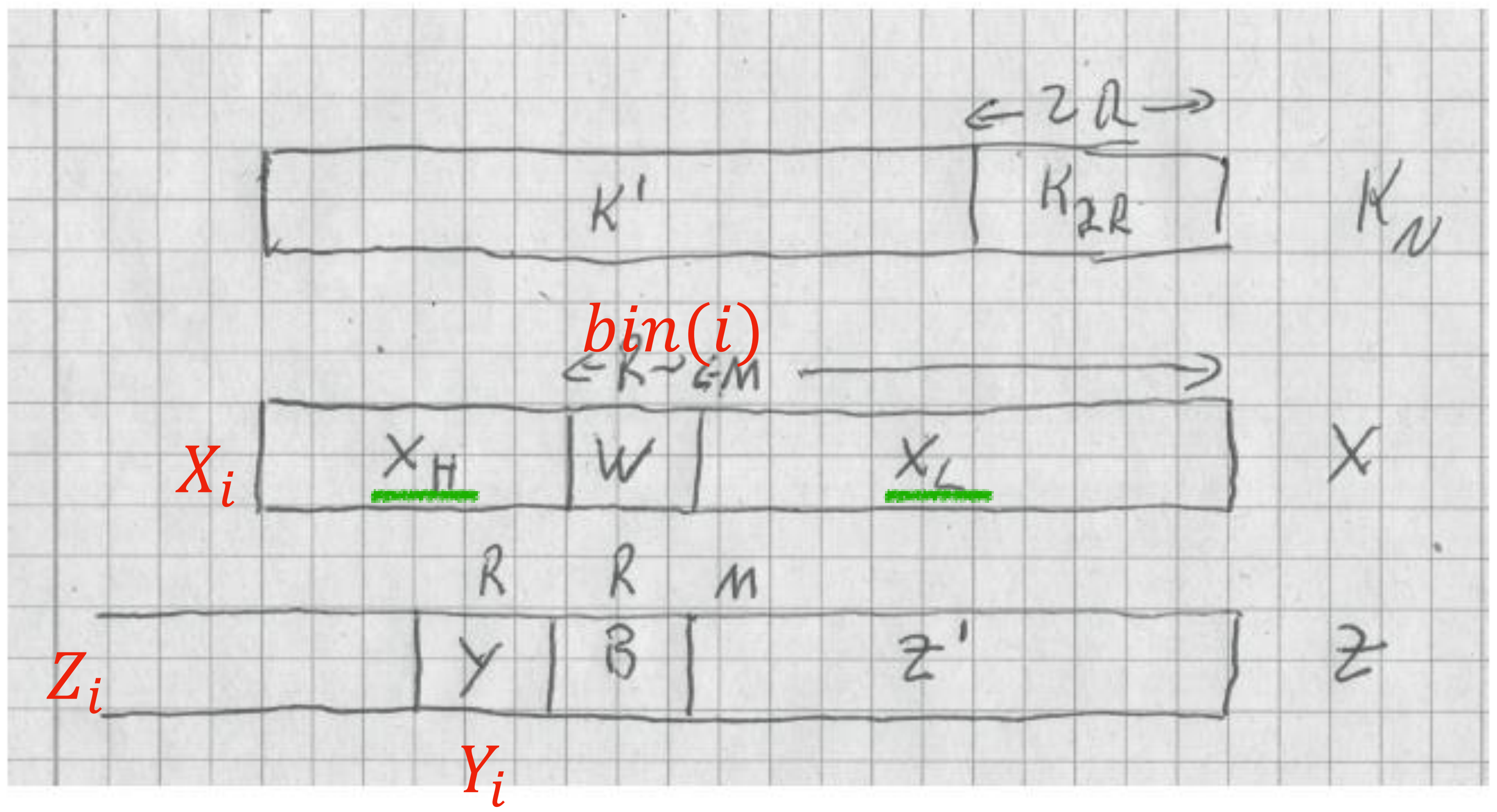
$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \\ z_j - z_i &\equiv (\langle B_j \rangle - \langle B_i \rangle) \cdot 2^M \pmod{2^{M+2R}} \\ (\langle B_j \rangle - \langle B_i \rangle) &= a \in \mathbb{Z} \quad , \quad |a| < 2^R \end{aligned}$$

$$\boxed{(j-i) \cdot k_{2R} \equiv a \pmod{2^{2R}}}$$

- $R = 2^r \rightarrow$

$$k_N = \langle K_N \rangle = \sum_{2^r < N} 2^{2^r}$$

$$2^R \leq 2^{2^0} + \ldots + 2^{2^{r-1}} + 2^{2^r} = k_{2R} \leq 2(2^R - 1)$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j-i) \cdot 2^M \cdot k_N \end{aligned}$$

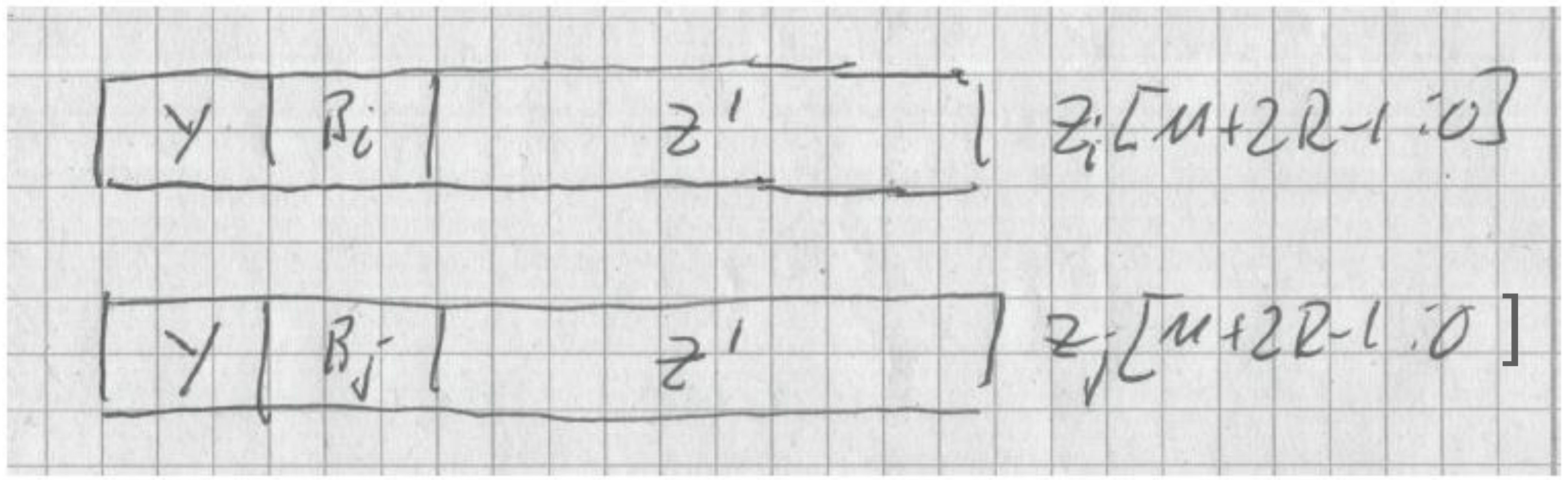
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j-i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \\ z_j - z_i &\equiv (\langle B_j \rangle - \langle B_i \rangle) \cdot 2^M \pmod{2^{M+2R}} \\ (\langle B_j \rangle - \langle B_i \rangle) &= a \in \mathbb{Z} \quad , \quad |a| < 2^R \\ &\boxed{(j-i) \cdot k_{2R} \equiv a \pmod{2^{2R}}} \end{aligned}$$

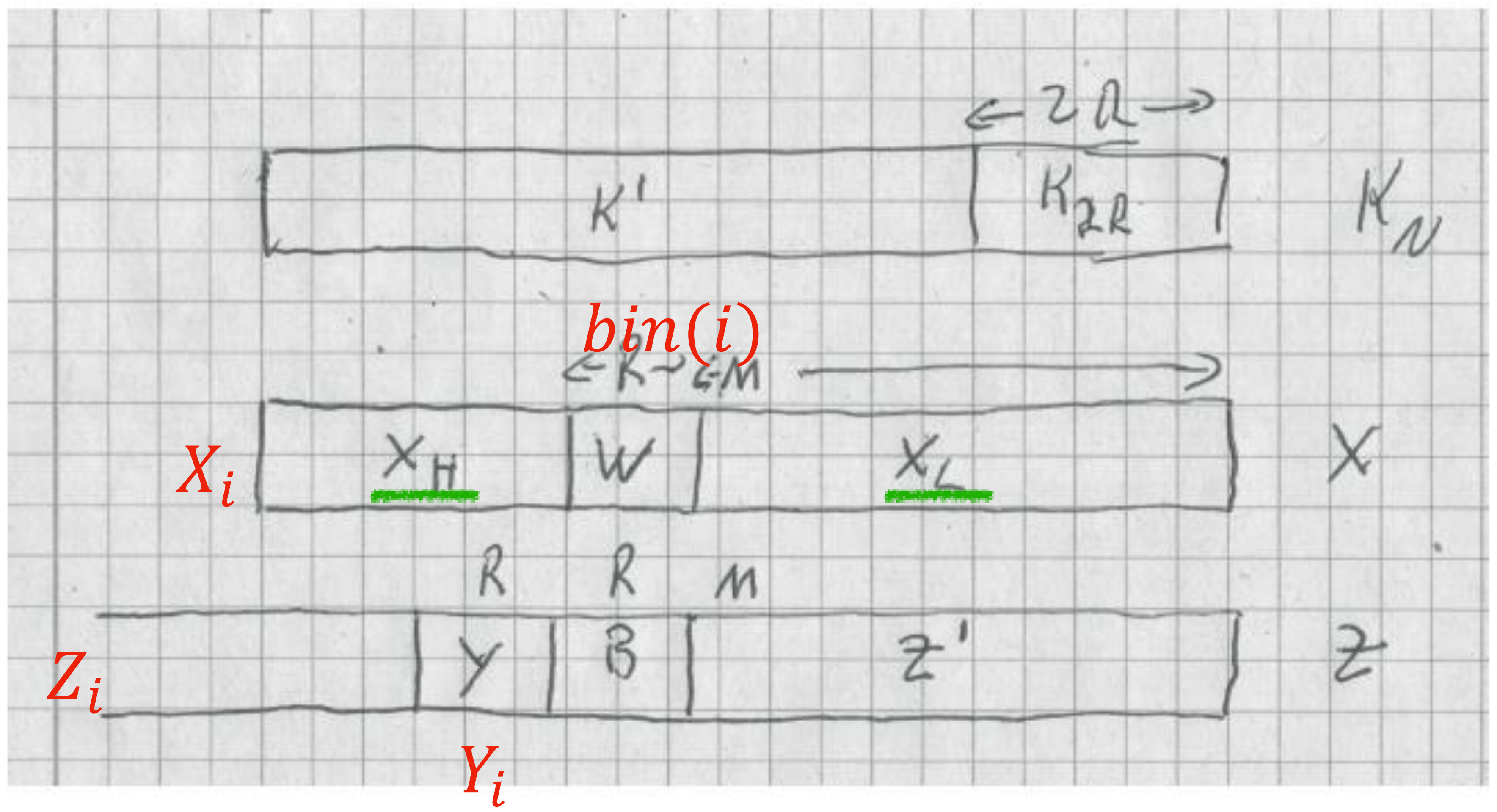
- $R = 2^r \rightarrow$

$$k_N = \langle K_N \rangle = \sum_{2^r < N} 2^{2^r}$$

$$2^R \leq 2^{2^0} + \ldots + 2^{2^{r-1}} + 2^{2^r} = k_{2R} \leq 2(2^R - 1)$$

-

$$(j-i) \cdot k_{2R} \geq 2^R > a$$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \ , \ y_i = \langle Y_i \rangle \ , \ z_i = \langle Z_i \rangle \ , k_i = \langle K_i \rangle \ , k' = \langle K' \rangle \ , \ k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j-i) \cdot 2^M \cdot k_N \end{aligned}$$

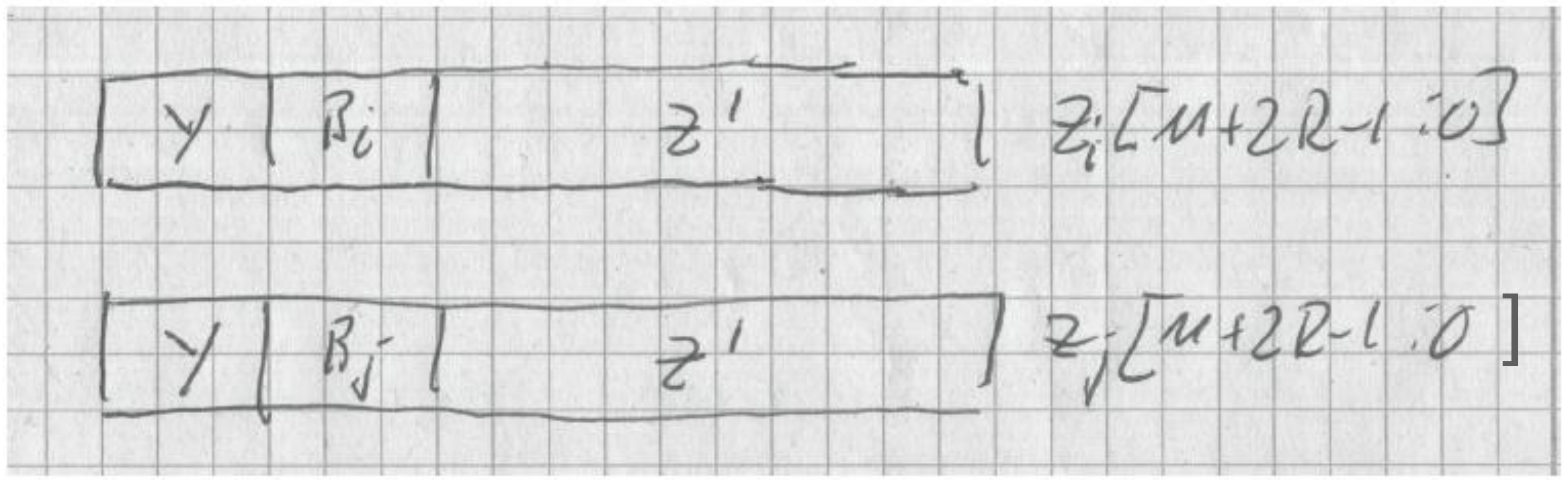
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j-i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \\ z_j - z_i &\equiv (\langle B_j \rangle - \langle B_i \rangle) \cdot 2^M \pmod{2^{M+2R}} \\ (\langle B_j \rangle - \langle B_i \rangle) &= a \in \mathbb{Z} \quad , \quad |a| < 2^R \end{aligned}$$

$$\boxed{(j-i) \cdot k_{2R} \equiv a \pmod{2^{2R}}}$$

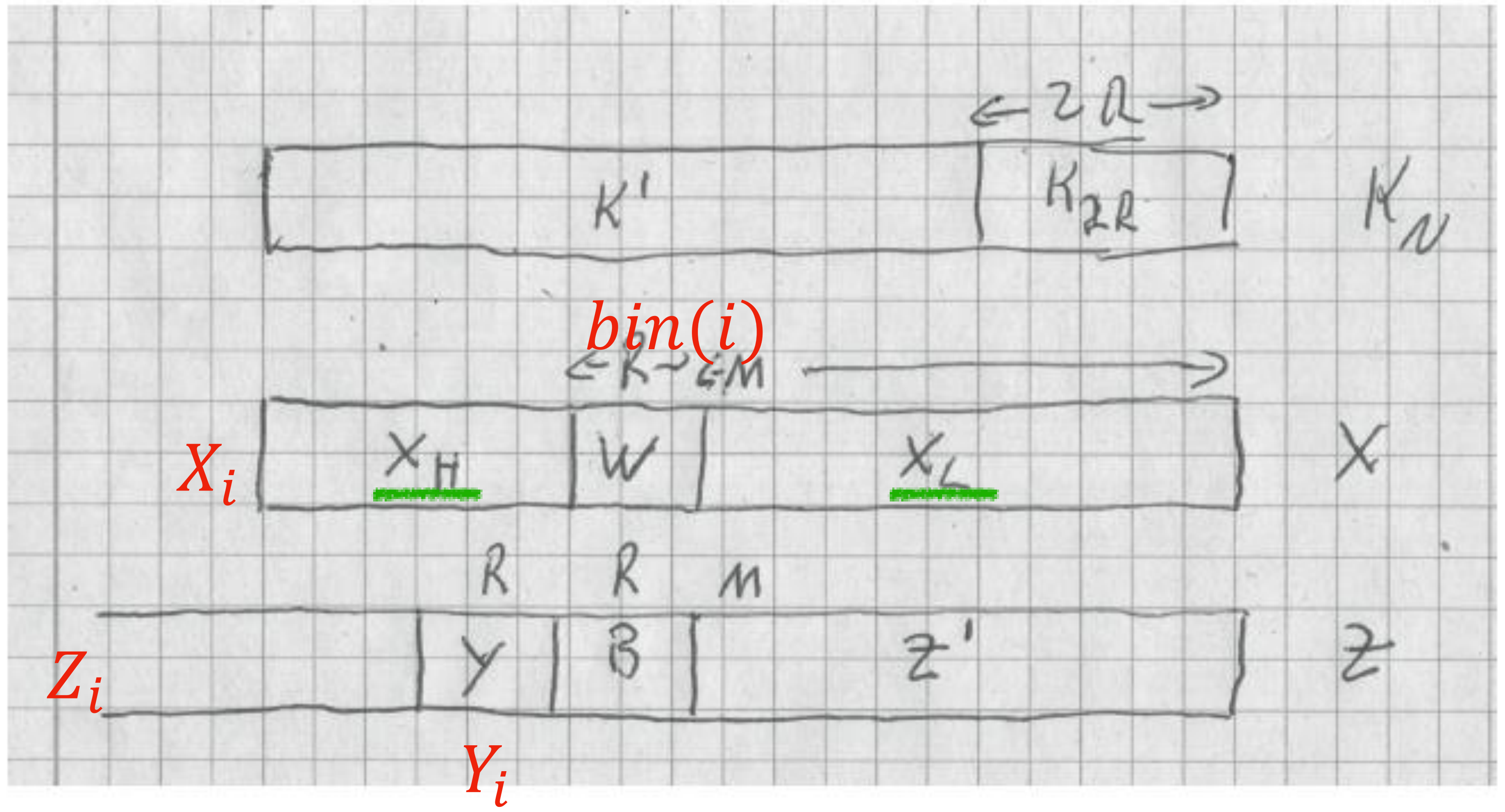
- $R = 2^r \rightarrow$ $\qquad k_N = \langle K_N \rangle = \sum_{2^r < N} 2^{2^r}$

$$2^R \leq 2^{2^0} + \ldots + 2^{2^{r-1}} + 2^{2^r} = k_{2R} \leq 2(2^R - 1)$$

-

$$(j-i) \cdot k_{2R} \geq 2^R > a$$

- for the congruence to hold: $\qquad (j-i) \cdot k_{2R} \geq a + 2^{2R} > 2^{2R} - 2^R$



Lemma 3. *For any given $Y \in \mathbb{B}^R$ there exist at most 2 indices i such that $Y_i = Y$, i.e. there are at most 2 possibilites for W.*

- abbreviate for all i

$$x_i = \langle X_i \rangle \;,\; y_i = \langle Y_i \rangle \;,\; z_i = \langle Z_i \rangle \;, k_i = \langle K_i \rangle \;, k' = \langle K' \rangle \;,\; k_{2R} = \langle K_{2R} \rangle$$

for $i < j$:

$$\begin{aligned} z_j - z_i &= (x_j - x_i) \cdot 2^M \cdot k_N \\ &= (j-i) \cdot 2^M \cdot k_N \end{aligned}$$

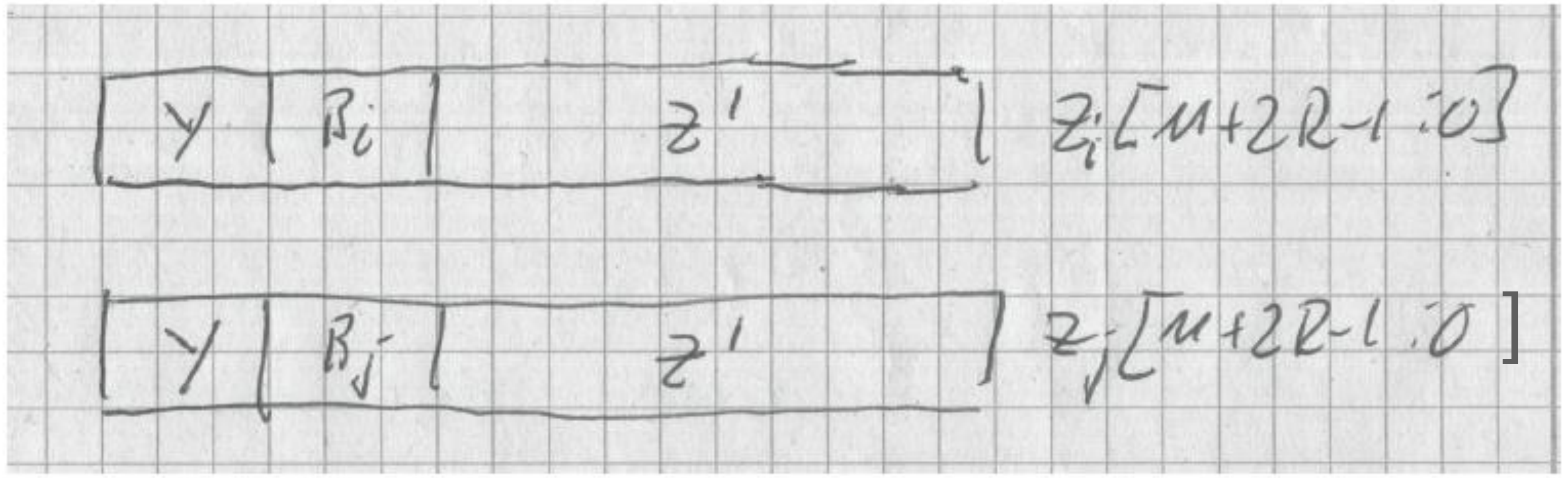
- $k_N = k_{2R} + k' \cdot 2^{2R} \rightarrow$

$$z_j - z_i \equiv (j-i) \cdot 2^M \cdot k_{2R} \pmod{2^{M+2R}}$$

- lemma 1 $\rightarrow$: low order bits of Z_i and Z_j are equal

$$Z_i[M-1:0] = Z_j[M-1:0] = Z'$$

- let $Y_i = Y_j = Y$ as shown in figure 2



$$\begin{aligned} Z_i[M+2R-1:0] &= Y \circ B_i \circ Z' \\ Z_j[M+2R-1:0] &= Y \circ B_j \circ Z' \\ z_j - z_i &\equiv (\langle B_j \rangle - \langle B_i \rangle) \cdot 2^M \pmod{2^{M+2R}} \\ (\langle B_j \rangle - \langle B_i \rangle) &= a \in \mathbb{Z} \quad , \quad |a| < 2^R \end{aligned}$$

$$\boxed{(j-i) \cdot k_{2R} \equiv a \pmod{2^{2R}}}$$

- $R = 2^r \rightarrow$ $\qquad k_N = \langle K_N \rangle = \sum_{2^r < N} 2^{2^r}$

$$2^R \le 2^{2^0} + \ldots + 2^{2^{r-1}} + 2^{2^r} = k_{2R} \le 2(2^R - 1)$$

-

$$(j-i) \cdot k_{2R} \ge 2^R > a$$

- for the congruence to hold: $\qquad (j-i) \cdot k_{2R} \ge a + 2^{2R} > 2^{2R} - 2^R$

$$\begin{aligned} j - i &\ge \frac{2^{2R} - 2^R}{2(2^R - 1)} \\ &= 2^R/2 \end{aligned}$$

4 Overlap argument

Let

- *Mult* be $t(N)$ time bounded k-tape online TM performing the multiplication

partitioning operands: For powers of two N and $i \in [0 : \log N - 1]$ partition operands $Y \in \mathbb{B}^N$ into $N/2^i$ intervals $Y_{i,j}$ of length 2^i as shown in figure 3.

$$Y_{i,j} = [(j+1) \cdot 2^i - 1 : j \cdot 2^i] \quad \text{for } j \in [0 : N/2^i - 1]$$

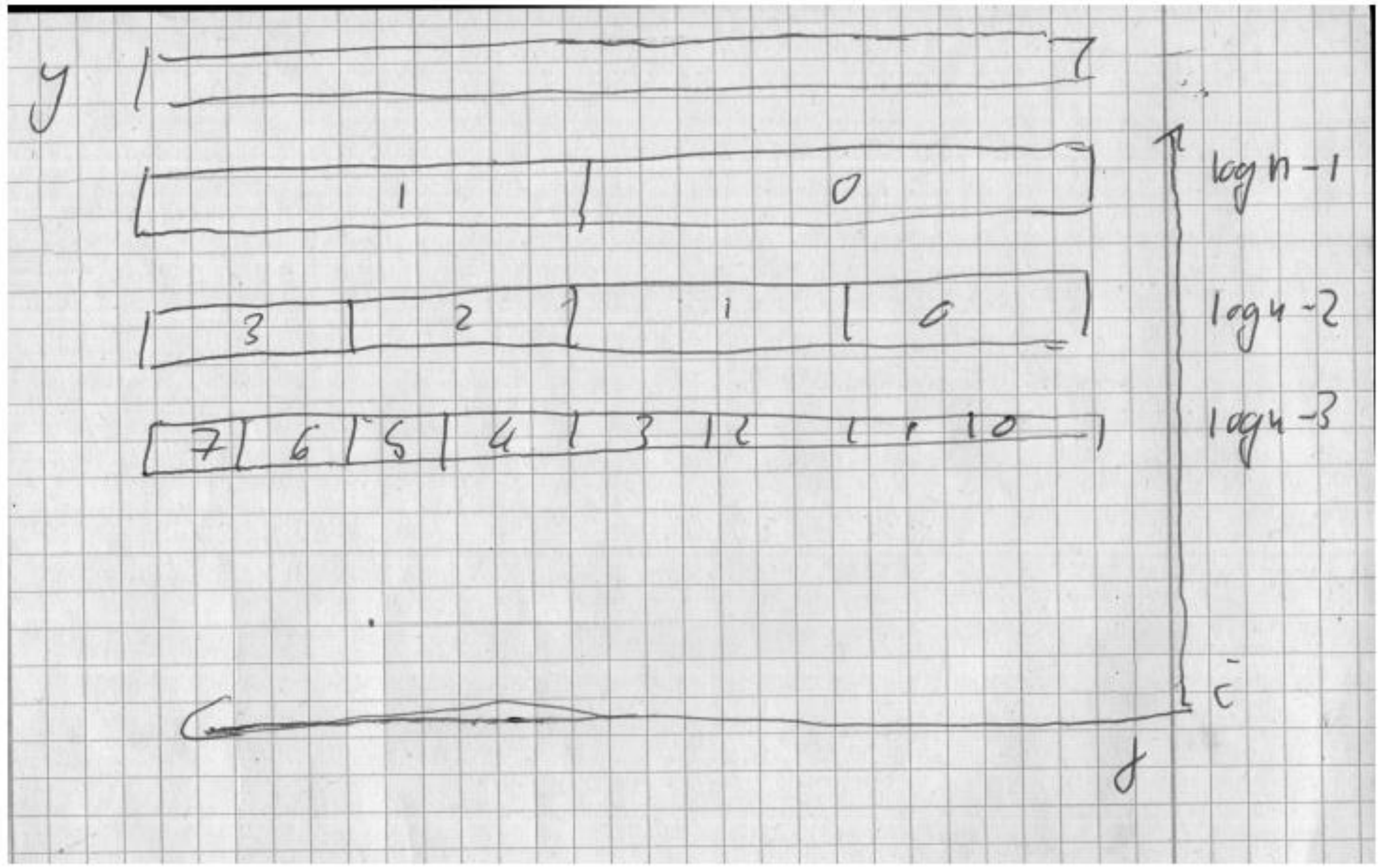


Figure 3: partitioning operands Y into intervals $Y_{i,j}$. The interval in row i indexed with j shows $Y_{i,j}$

4 Overlap argument

Let

- *Mult* be $t(N)$ time bounded k-tape online TM performing the multiplication

partitioning operands: For powers of two N and $i \in [0 : \log N - 1]$ partition operands $Y \in \mathbb{B}^N$ into $N/2^i$ intervals $Y_{i,j}$ of length 2^i as shown in figure 3.

$$Y_{i,j} = [(j+1) \cdot 2^i - 1 : j \cdot 2^i] \quad \text{for } j \in [0 : N/2^i - 1]$$

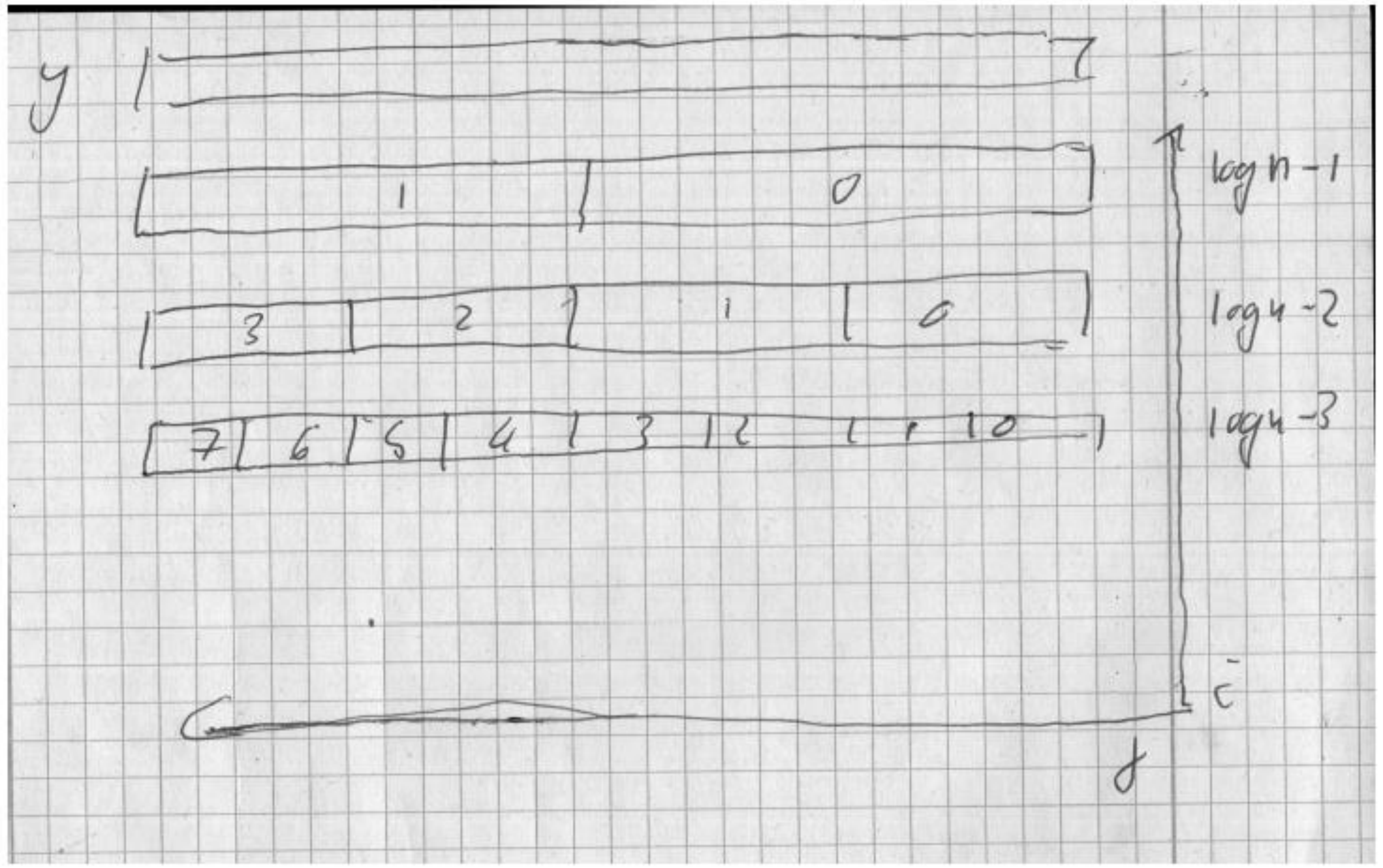


Figure 3: partitioning operands Y into intervals $Y_{i,j}$. The interval in row i indexed with j shows $Y_{i,j}$

time intervals $T_{i,j}$: The steps to process $X_{i,j}$ and $K_{i,j}$ (from reading the first pair of input bits to printing the last output bit).

4 Overlap argument

Let

- *Mult* be $t(N)$ time bounded k-tape online TM performing the multiplication

partitioning operands: For powers of two N and $i \in [0 : \log N - 1]$ partition operands $Y \in \mathbb{B}^N$ into $N/2^i$ intervals $Y_{i,j}$ of length 2^i as shown in figure 3.

$$Y_{i,j} = [(j+1) \cdot 2^i - 1 : j \cdot 2^i] \quad \text{for } j \in [0 : N/2^i - 1]$$

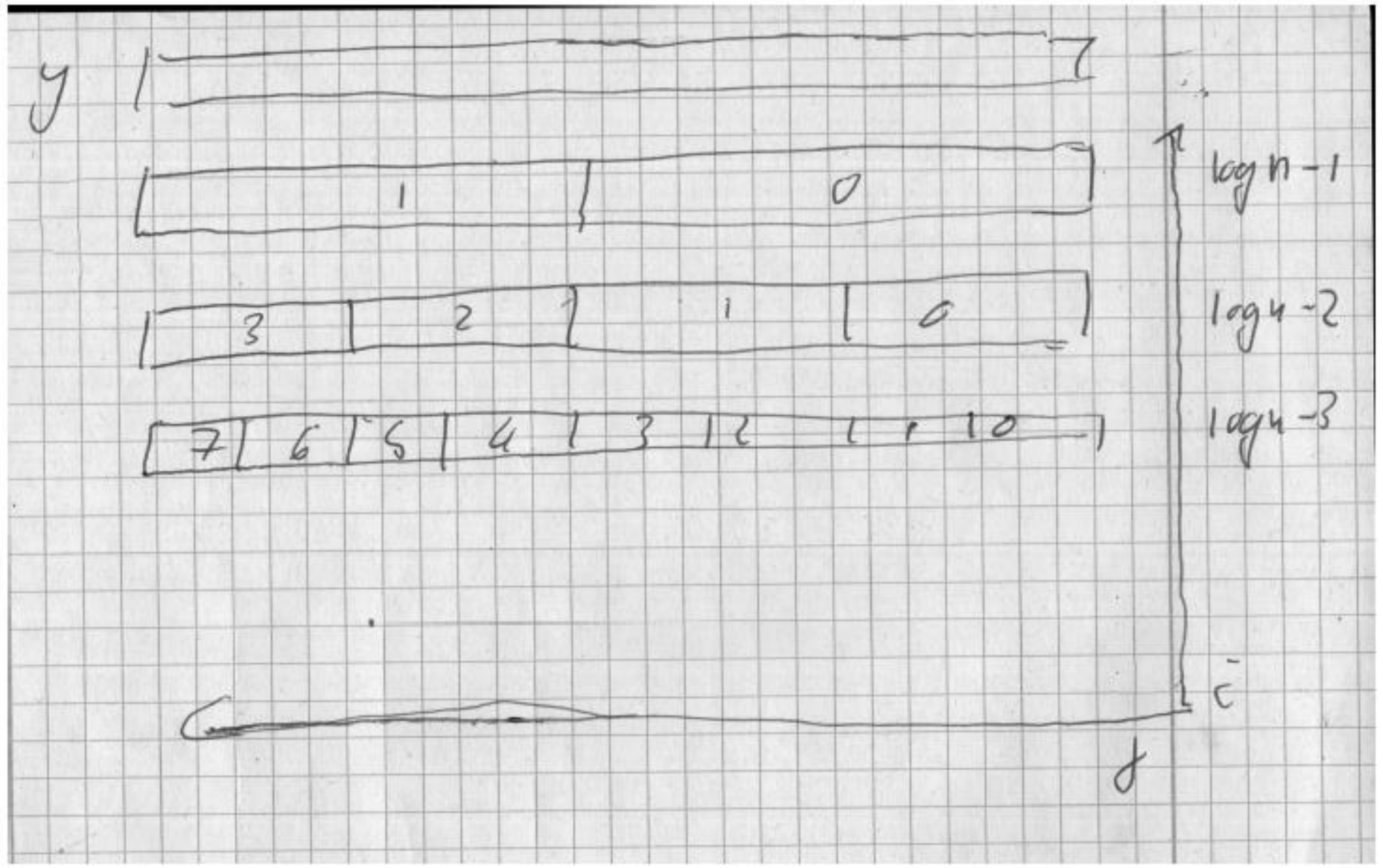


Figure 3: partitioning operands Y into intervals $Y_{i,j}$. The interval in row i indexed with j shows $Y_{i,j}$

time intervals $T_{i,j}$: The steps to process $X_{i,j}$ and $K_{i,j}$ (from reading the first pair of input bits to printing the last output bit).

computation graphs for $t(N)$ steps and work tape τ:

- nodes: the time steps

$$V = [0 : t(n)]$$

- edges $(t,t') \in E^\tau$ from t to t' iff $t' > t$ and there is a tape cell of tape τ visitend in steps t an t' but not in between.

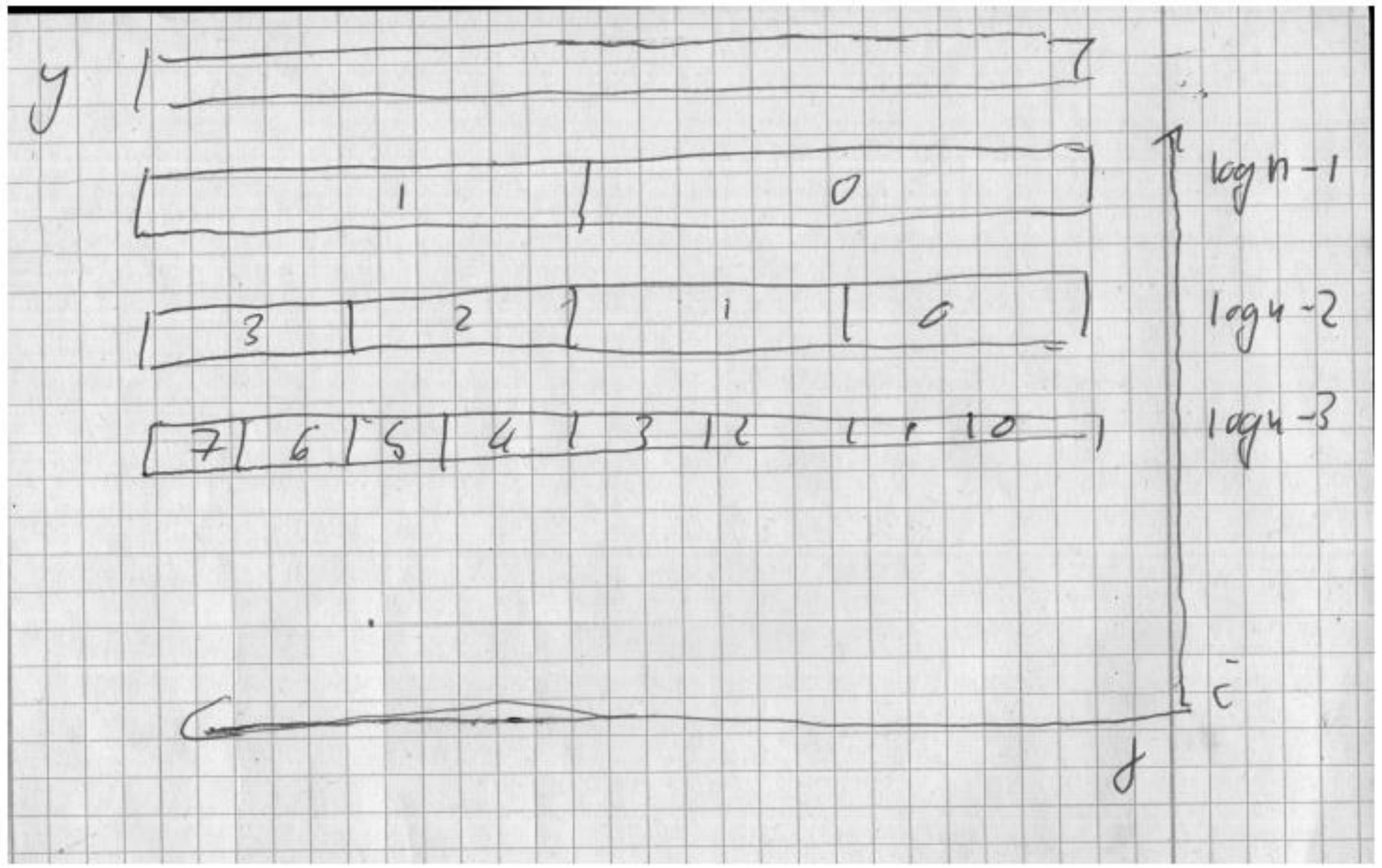


Figure 3: partitioning operands Y into intervals $Y_{i,j}$. The interval in row i indexed with j shows $Y_{i,j}$

time intervals $T_{i,j}$: The steps to process $X_{i,j}$ and $K_{i,j}$ (from reading the first pair of input bits to printing the last output bit).

computation graphs for $t(N)$ steps and work tape τ:

- nodes: the time steps
$$V = [0 : t(n)]$$

- edges $(t,t') \in E^{\tau}$ from t to t' iff $t' > t$ and there is a tape cell of tape τ visitend in steps t an t' but not in between.

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

-
$$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

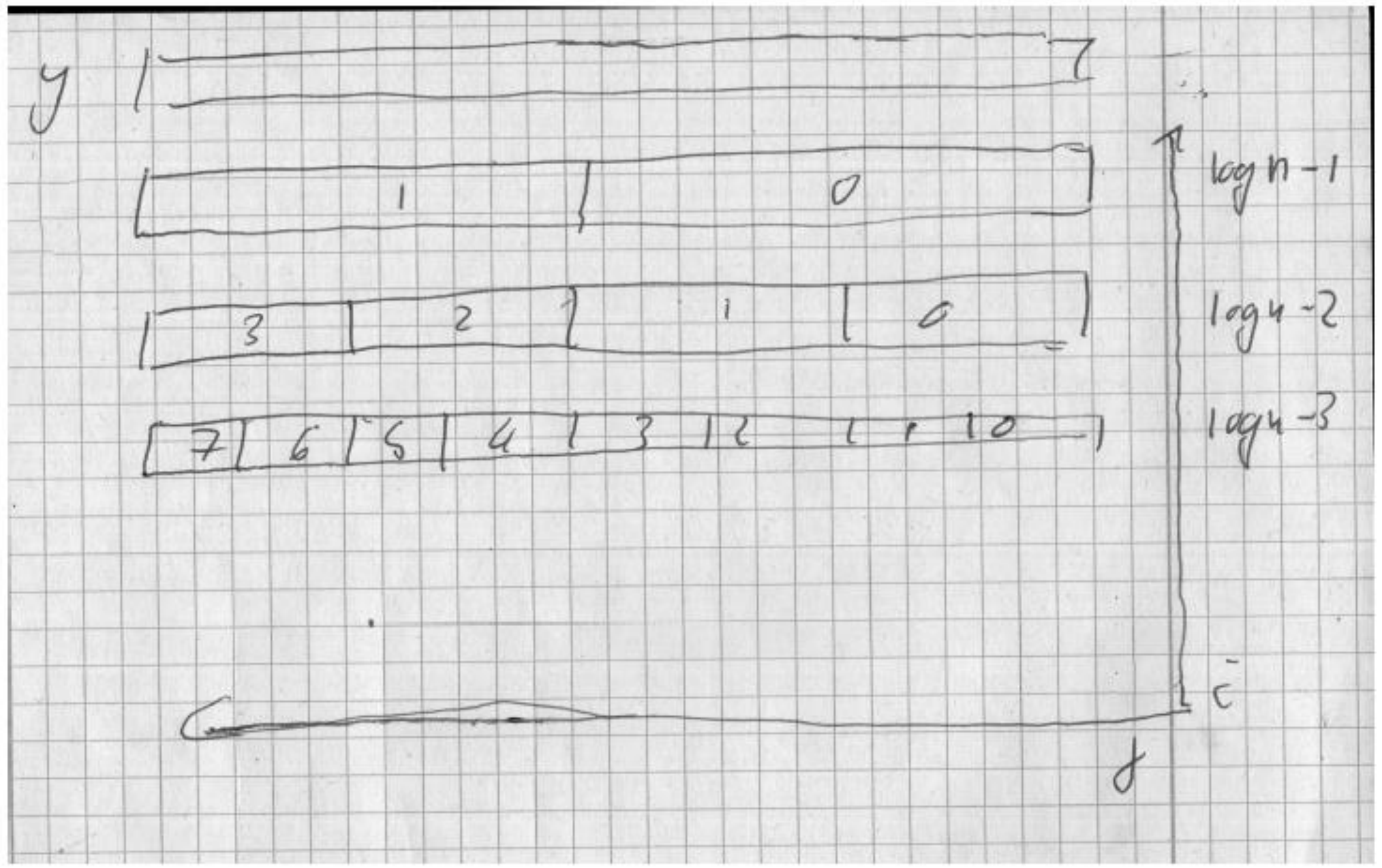


Figure 3: partitioning operands Y into intervals $Y_{i,j}$. The interval in row i indexed with j shows $Y_{i,j}$

time intervals $T_{i,j}$: The steps to process $X_{i,j}$ and $K_{i,j}$ (from reading the first pair of input bits to printing the last output bit).

computation graphs for $t(N)$ steps and work tape τ:

- nodes: the time steps

$$V = [0 : t(n)]$$

- edges $(t,t') \in E^\tau$ from t to t' iff $t' > t$ and there is a tape cell of tape τ visitend in steps t an t' but not in between.

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^\tau_{i,2j} = E^\tau \cap (T_{i,2j} \times T_{i,2j+1})$$

- $$E^\tau = \dot{\bigcup}_i \dot{\bigcup}_j E^\tau_{i,2j}$$

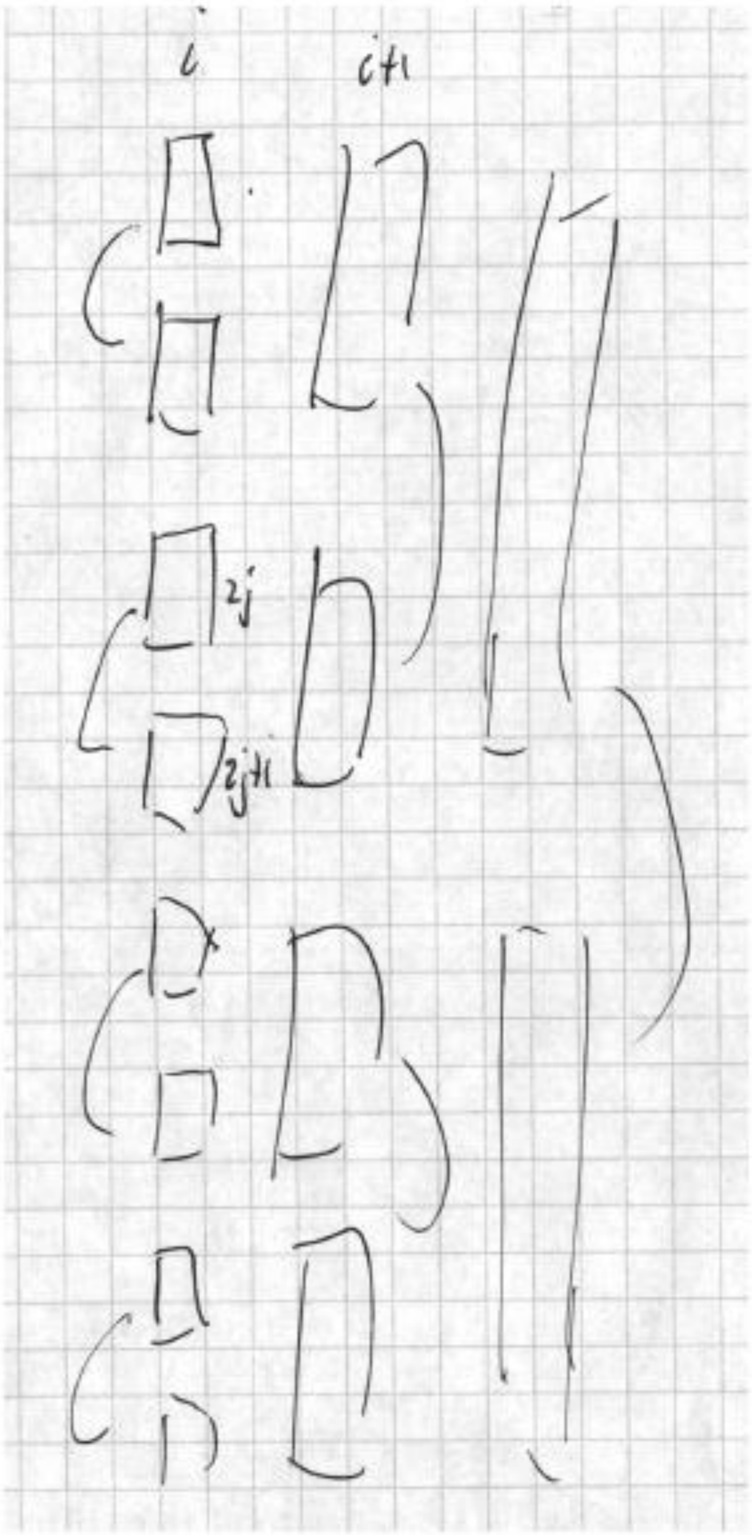


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

•

$$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

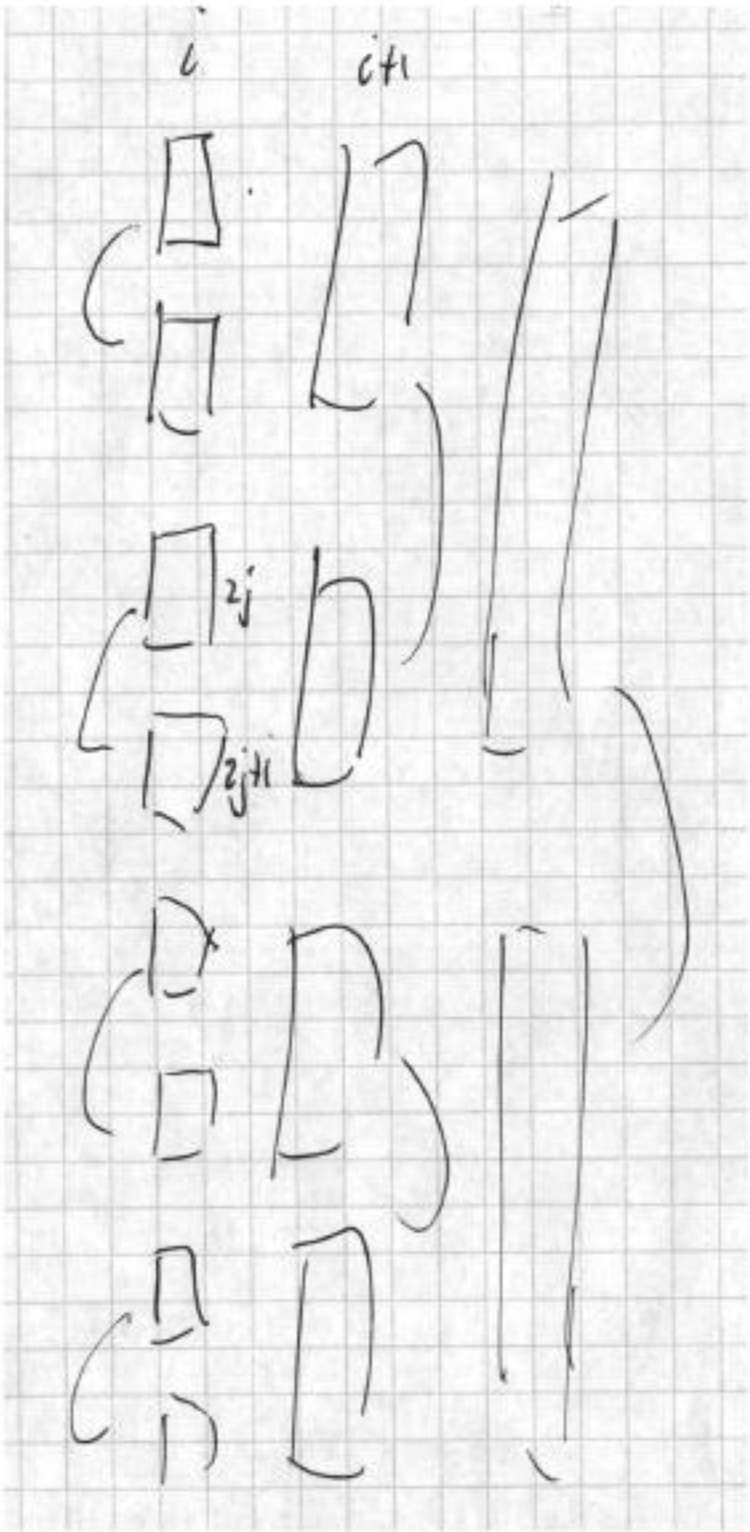


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

- $$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

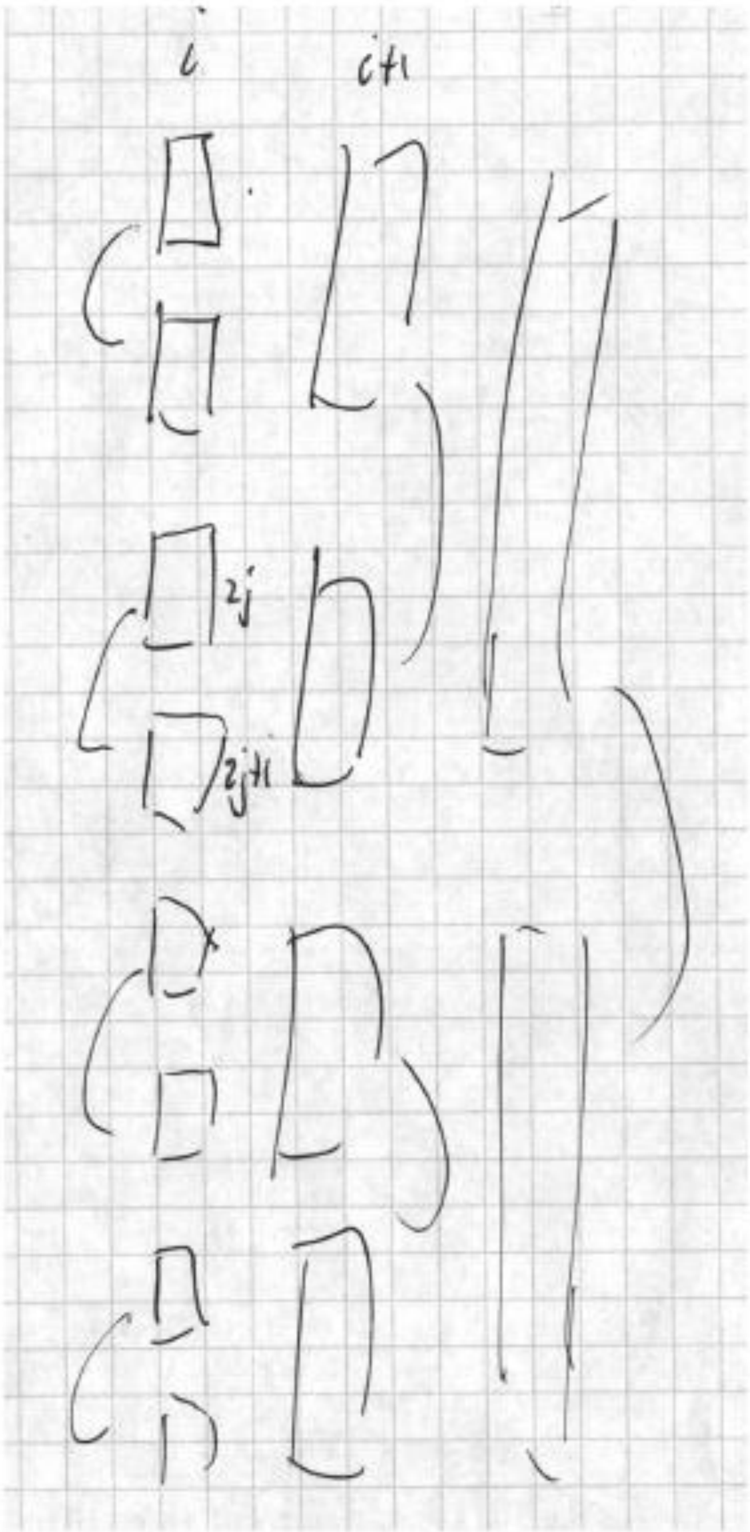


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

- $$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

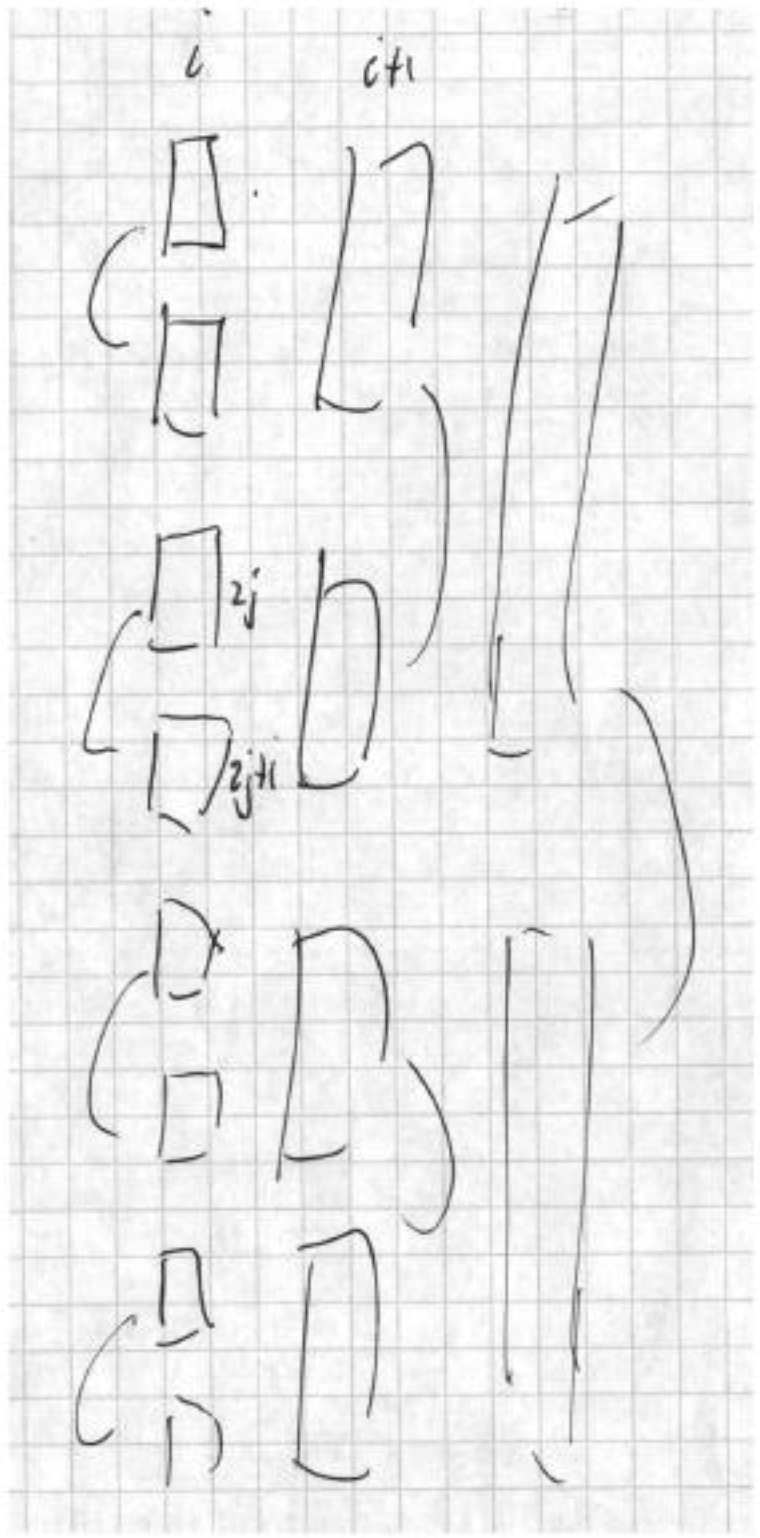


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

def: overlap $\omega^{\tau}_{i,2j}$ on a tape : the portion of tape τ visited in time interval $T_{i,2j}$ and revisited in time inteval $T_{i,2j+1}$.

- on a single tape τ an edge in $E^{\tau}_{i,2j}$ can contribute at most 1 tape cell to overlap

$$|\omega^{\tau}_{i,2j}| \leq |E^{\tau}_{i,2j}| \leq T$$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

- $$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

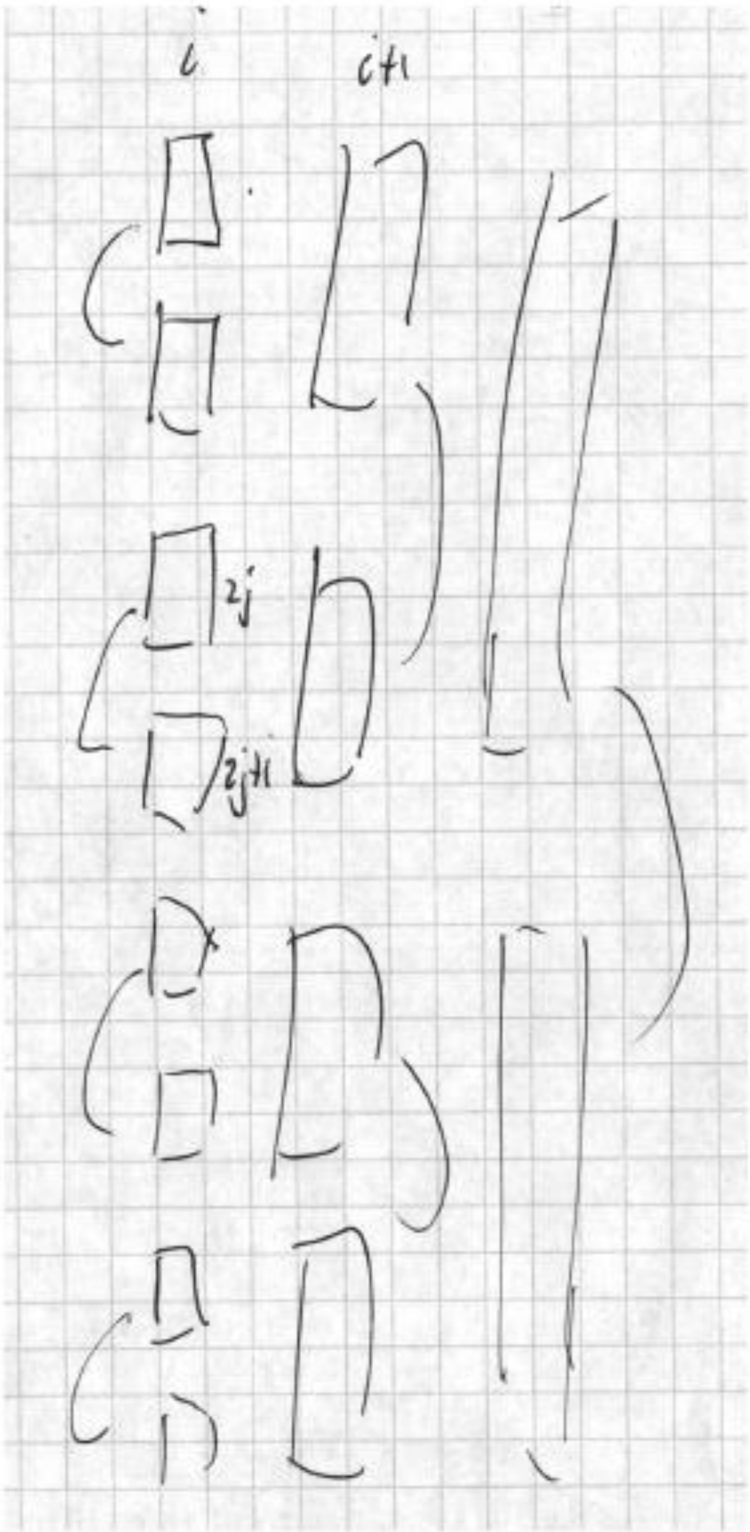


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

def: overlap $\omega^{\tau}_{i,2j}$ on a tape : the portion of tape τ visited in time interval $T_{i,2j}$ and revisited in time inteval $T_{i,2j+1}$.

- on a single tape τ an edge in $E^{\tau}_{i,2j}$ can contribute at most 1 tape cell to overlap

$$|\omega^{\tau}_{i,2j}| \leq |E^{\tau}_{i,2j}| \leq T$$

result $res(i,2j)$ of time interval $T_{i,2j}$: contains for each tape τ

- the position of the overlap region $\omega^{\tau}_{i,2j}$ on the tape. It is an interval, so it suffices to specify the borders

$$O(\log(t(N)) \text{ bits}$$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

•

$$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

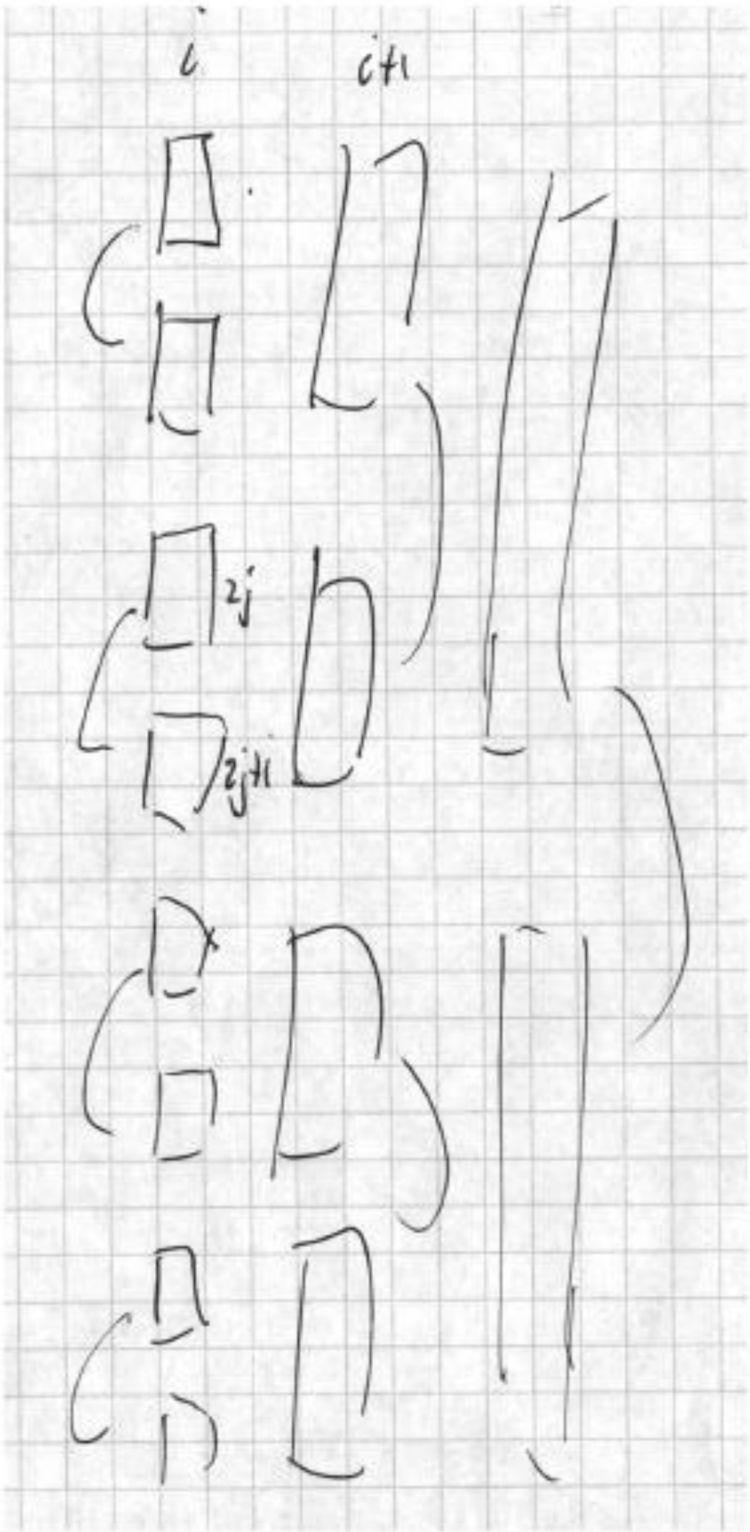


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

def: overlap $\omega^{\tau}_{i,2j}$ on a tape : the portion of tape τ visited in time interval $T_{i,2j}$ and revisited in time inteval $T_{i,2j+1}$.

- on a single tape τ an edge in $E^{\tau}_{i,2j}$ can contribute at most 1 tape cell to overlap

$$|\omega^{\tau}_{i,2j}| \leq |E^{\tau}_{i,2j}| \leq T$$

result $res(i,2j)$ of time interval $T_{i,2j}$: contains for each tape τ

- the position of the overlap region $\omega^{\tau}_{i,2j}$ on the tape. It is an interval, so it suffices to specify the borders

$$O(\log(t(N)) \text{ bits}$$

- for all tapes τ inscription of the overlap region and head position at the end of time interval $T_{i,2j}$.

$$c \cdot |\omega^{\tau}_{i,2j}| + \log(t(N)) \text{ bits for some } c > 0$$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

- $$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

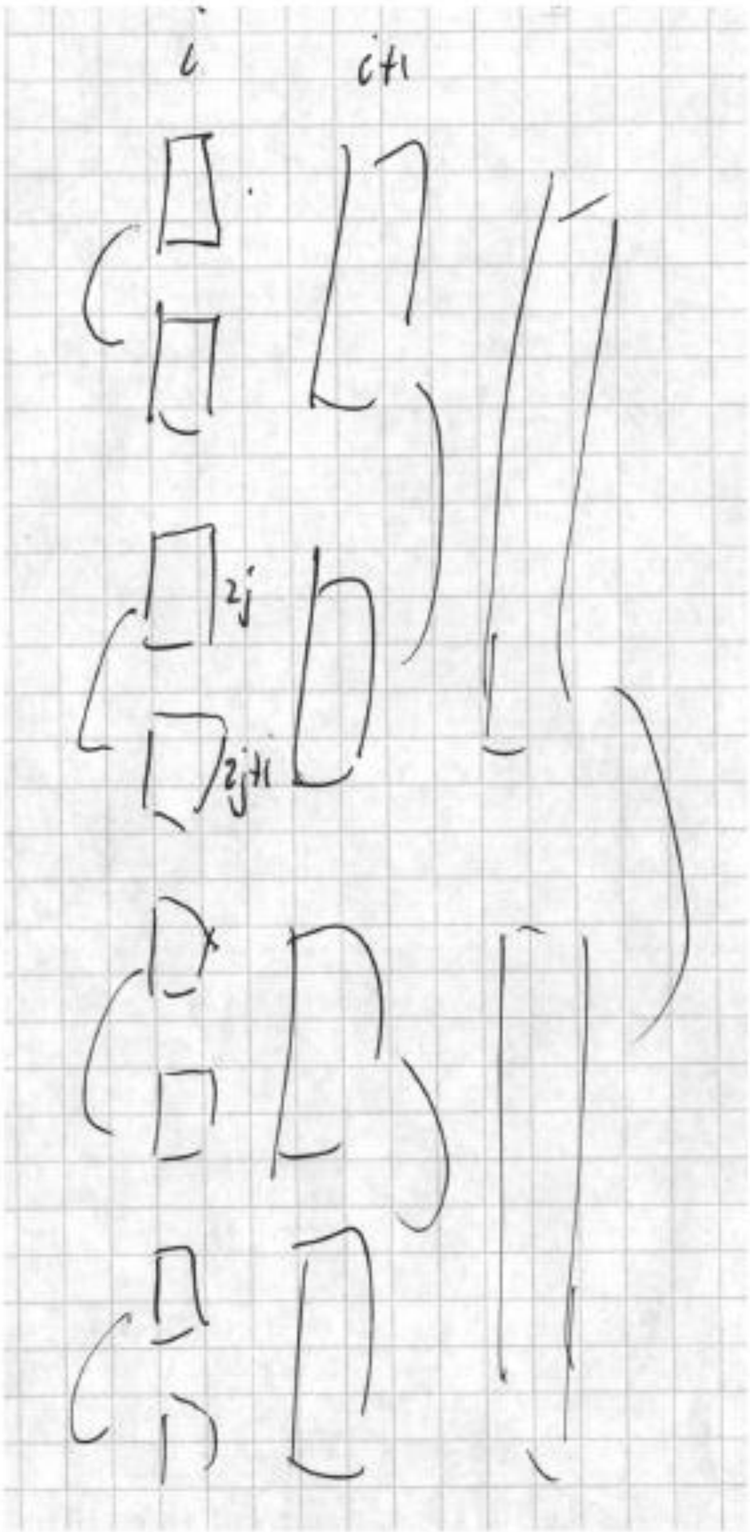


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

def: overlap $\omega^{\tau}_{i,2j}$ on a tape : the portion of tape τ visited in time interval $T_{i,2j}$ and revisited in time inteval $T_{i,2j+1}$.

- on a single tape τ an edge in $E^{\tau}_{i,2j}$ can contribute at most 1 tape cell to overlap

$$|\omega^{\tau}_{i,2j}| \leq |E^{\tau}_{i,2j}| \leq T$$

result $res(i,2j)$ of time interval $T_{i,2j}$: contains for each tape τ

- the position of the overlap region $\omega^{\tau}_{i,2j}$ on the tape. It is an interval, so it suffices to specify the borders

$$O(\log(t(N)) \text{ bits}$$

- for all tapes τ inscription of the overlap region and head position at the end of time interval $T_{i,2j}$.

$$c \cdot |\omega^{\tau}_{i,2j}| + \log(t(N)) \text{ bits for some } c > 0$$

- state of $Mult$ at the end of time interval $T_{i,2j}$

$$O(1) \text{ bits}$$

Partitioning edges into classes $E_{i,2j}$: the edges from time interval $T_{i,2j}$ to the next time interval $T_{i,2j+1}$ as shown in figure 4.

$$E^{\tau}_{i,2j} = E^{\tau} \cap (T_{i,2j} \times T_{i,2j+1})$$

•

$$E^{\tau} = \dot{\bigcup}_i \dot{\bigcup}_j E^{\tau}_{i,2j}$$

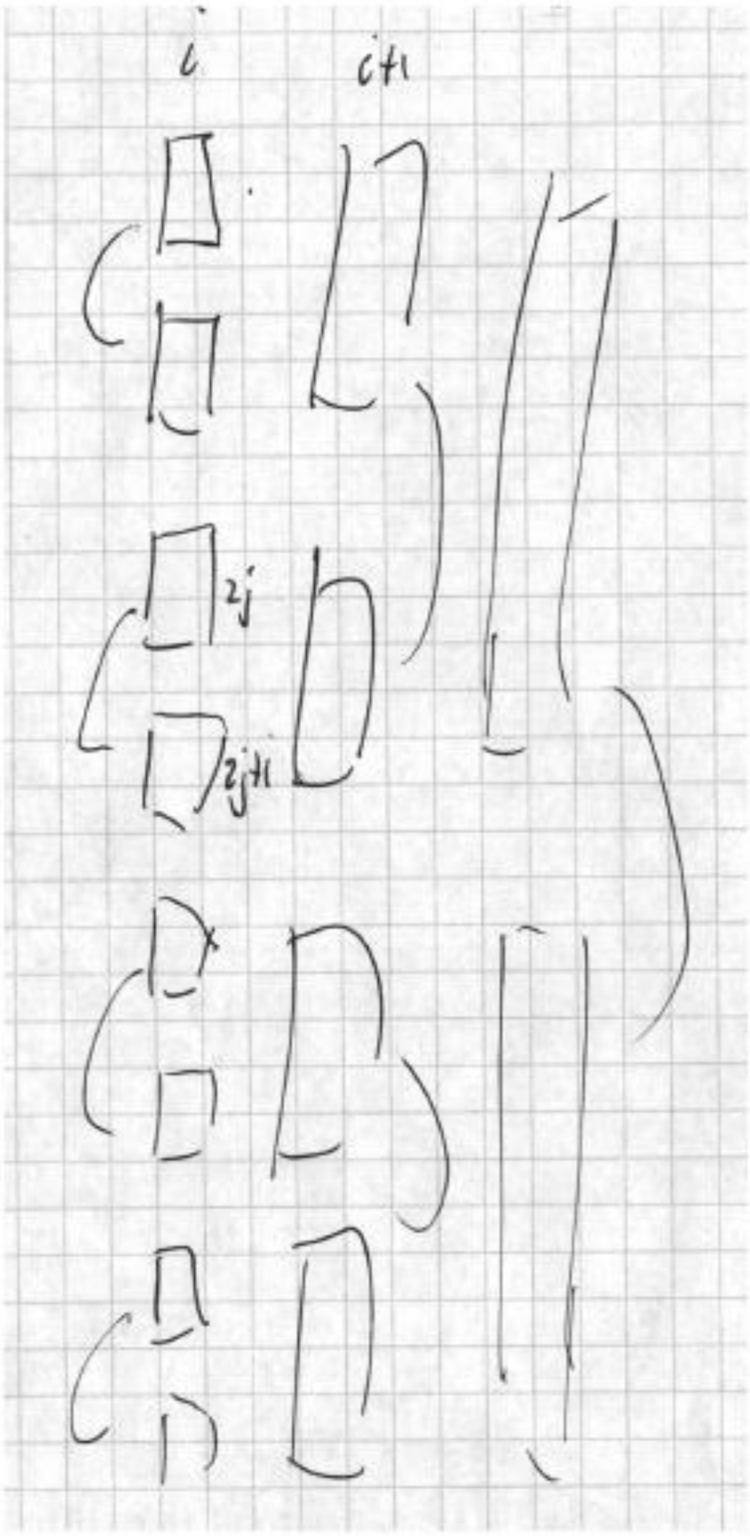


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

- at most one edge per step on each tape τ

$$\boxed{T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|}$$

def: overlap $\omega^{\tau}_{i,2j}$ on a tape : the portion of tape τ visited in time interval $T_{i,2j}$ and revisited in time inteval $T_{i,2j+1}$.

- on a single tape τ an edge in $E^{\tau}_{i,2j}$ can contribute at most 1 tape cell to overlap

$$|\omega^{\tau}_{i,2j}| \leq |E^{\tau}_{i,2j}| \leq T$$

result $res(i,2j)$ of time interval $T_{i,2j}$: contains for each tape τ

- the position of the overlap region $\omega^{\tau}_{i,2j}$ on the tape. It is an interval, so it suffices to specify the borders

$$O(\log(t(N)) \text{ bits}$$

- for all tapes τ inscription of the overlap region and head position at the end of time interval $T_{i,2j}$.

$$c \cdot |\omega^{\tau}_{i,2j}| + \log(t(N)) \text{ bits for some } c > 0$$

- state of $Mult$ at the end of time interval $T_{i,2j}$

$$O(1) \text{ bits}$$

- with size of overlap region $\omega_{i,2j}$ for all tapes

$$\boxed{|\omega_{i,2j}| = \sum_{\tau} |\omega^{\tau}_{i,2j}|}$$

we get

$$\boxed{|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}}$$

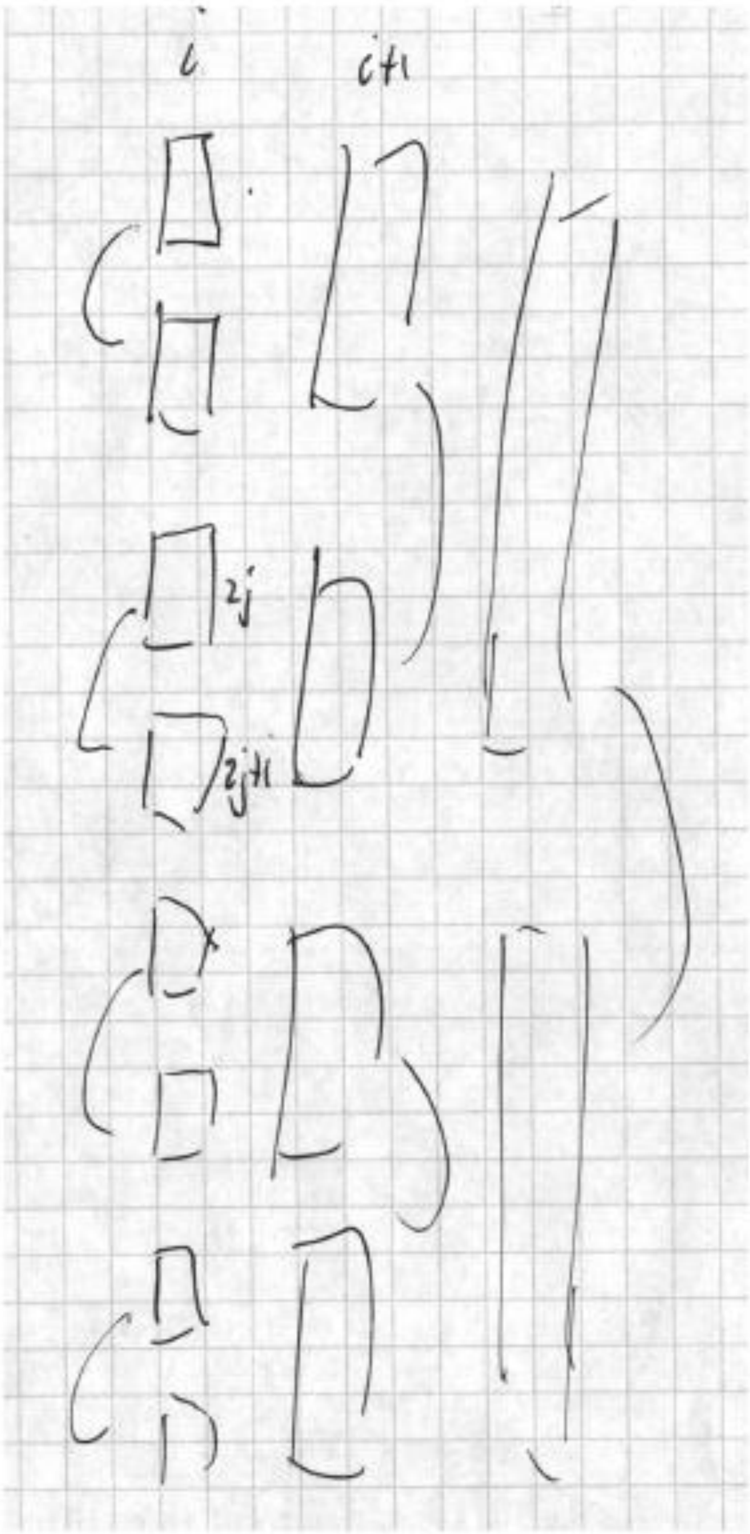


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

$$|\omega_{i,2j}| = \sum_{\tau} |\omega^{\tau}_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}$$

5 Incompressibility argument

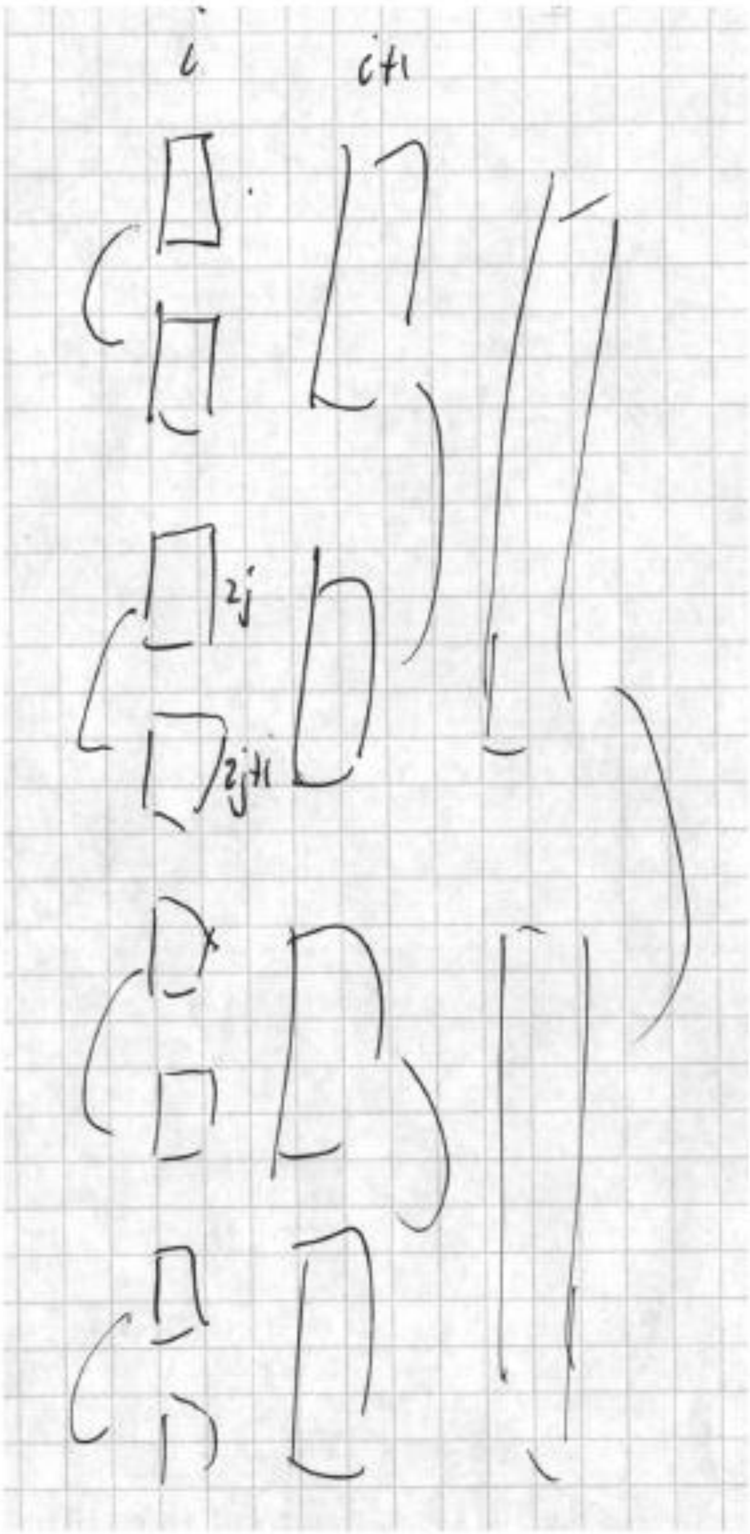
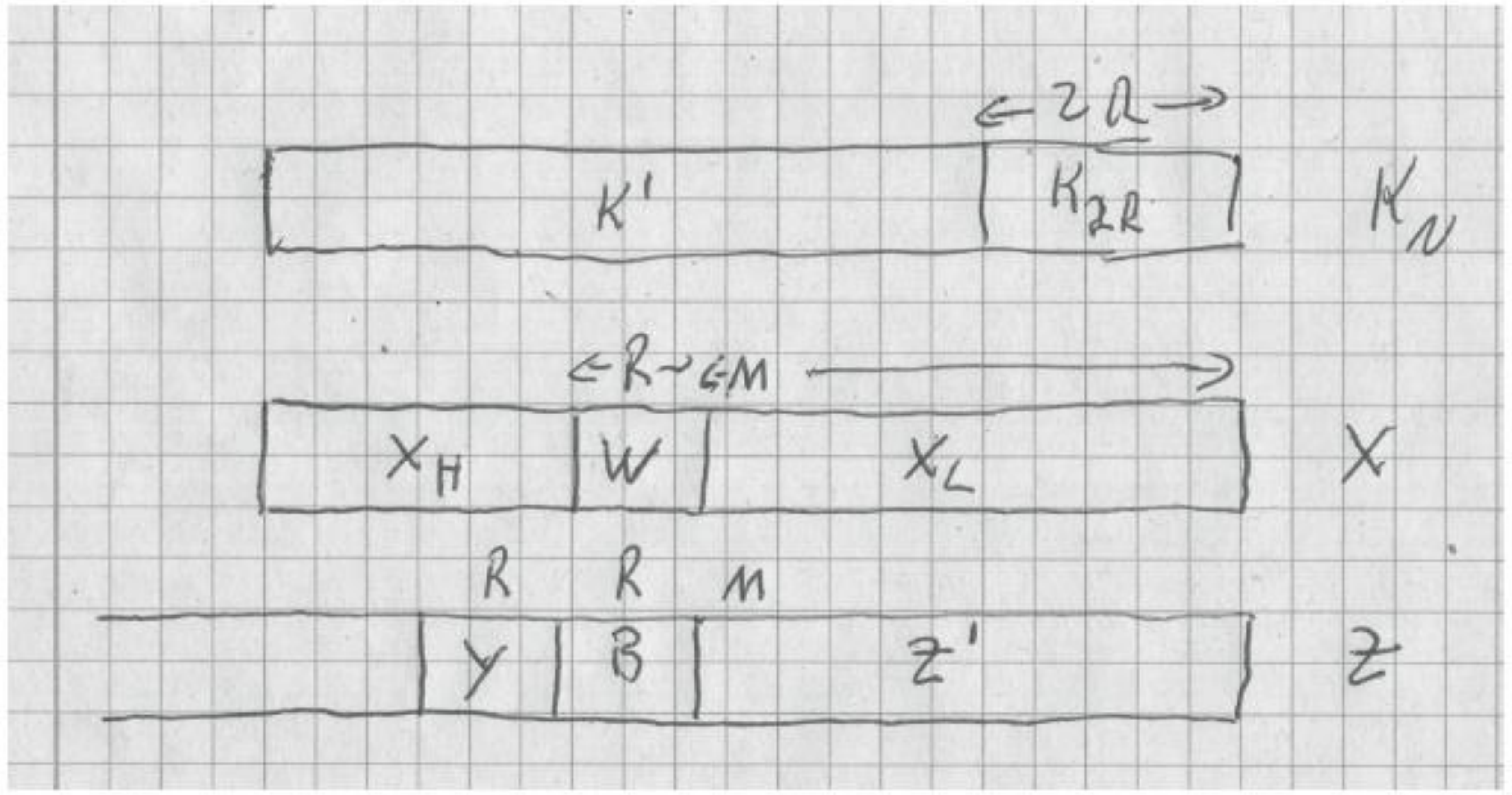


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

$$|\omega_{i,2j}| = \sum_{\tau} |\omega^{\tau}_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N))) \text{ bits}$$



choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1

5 Incompressibility argument

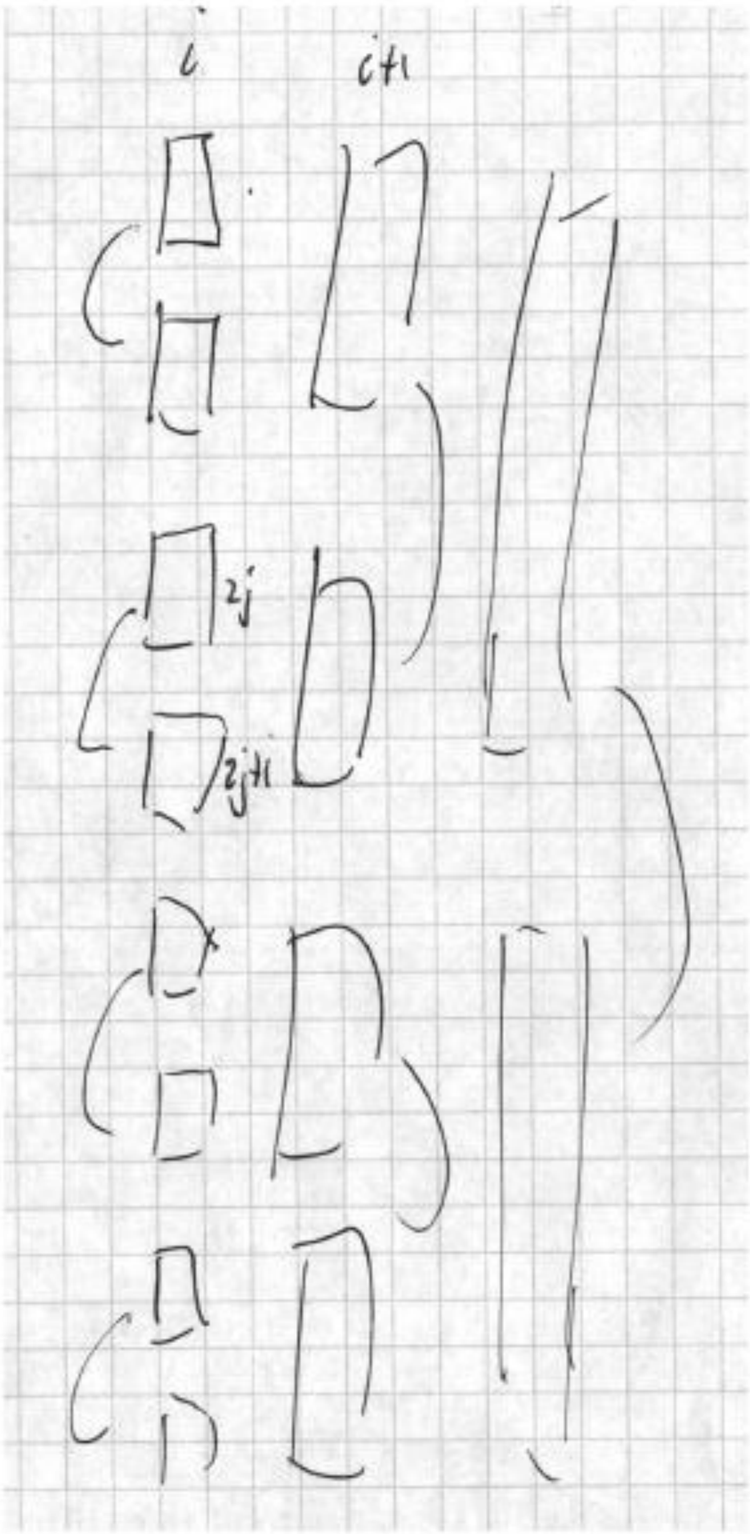
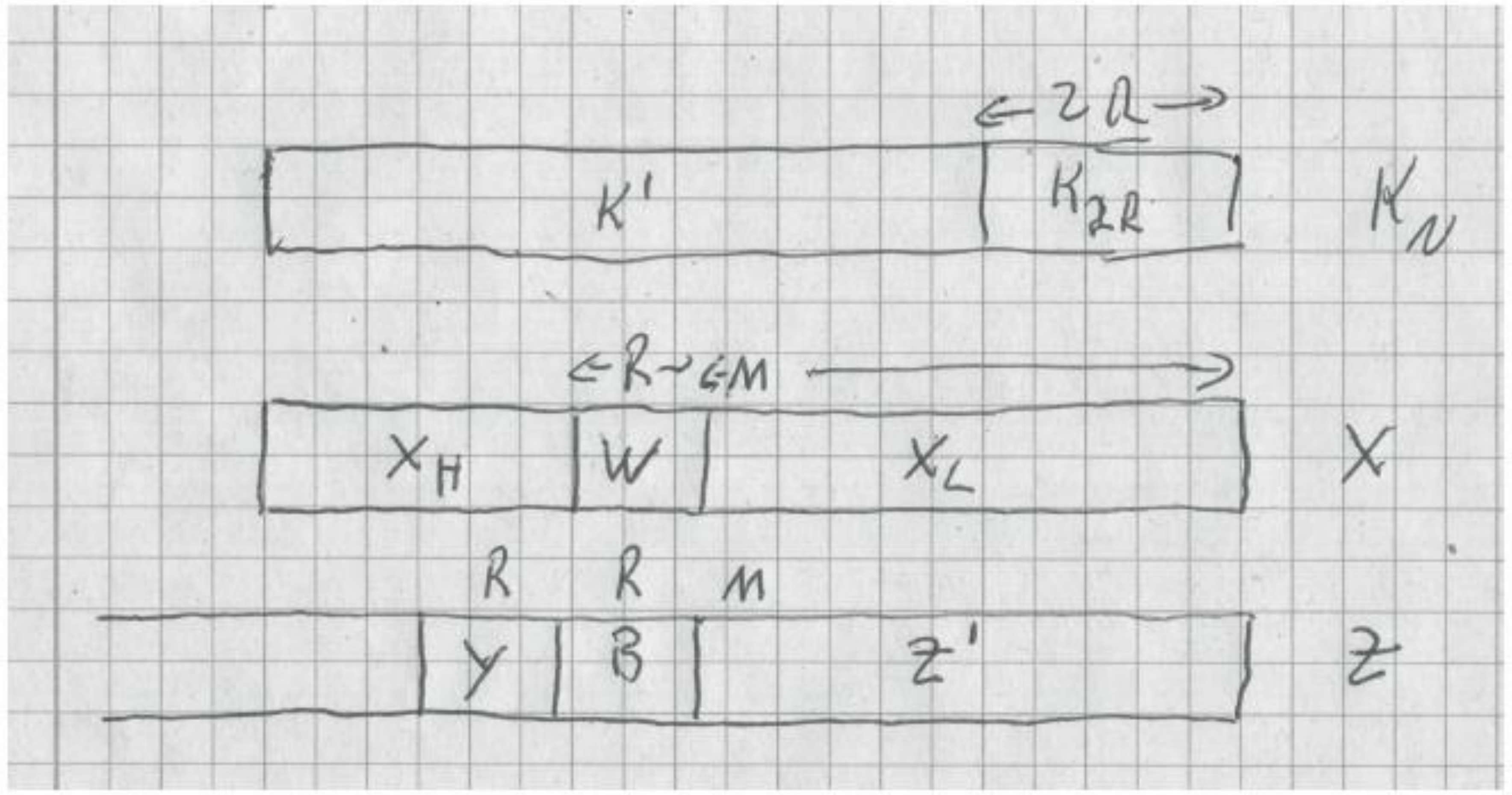


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

$$|\omega_{i,2j}| = \sum_{\tau} |\omega^{\tau}_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}$$



choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1
- substrings W of random strings X are locally almost random, even given the remainder $X_H X_L$ of the string

$$K(W|\, X_H X_L) \geq |Y| - O(\log(N)) = 2^i - O(\log(N))$$

shown earlier

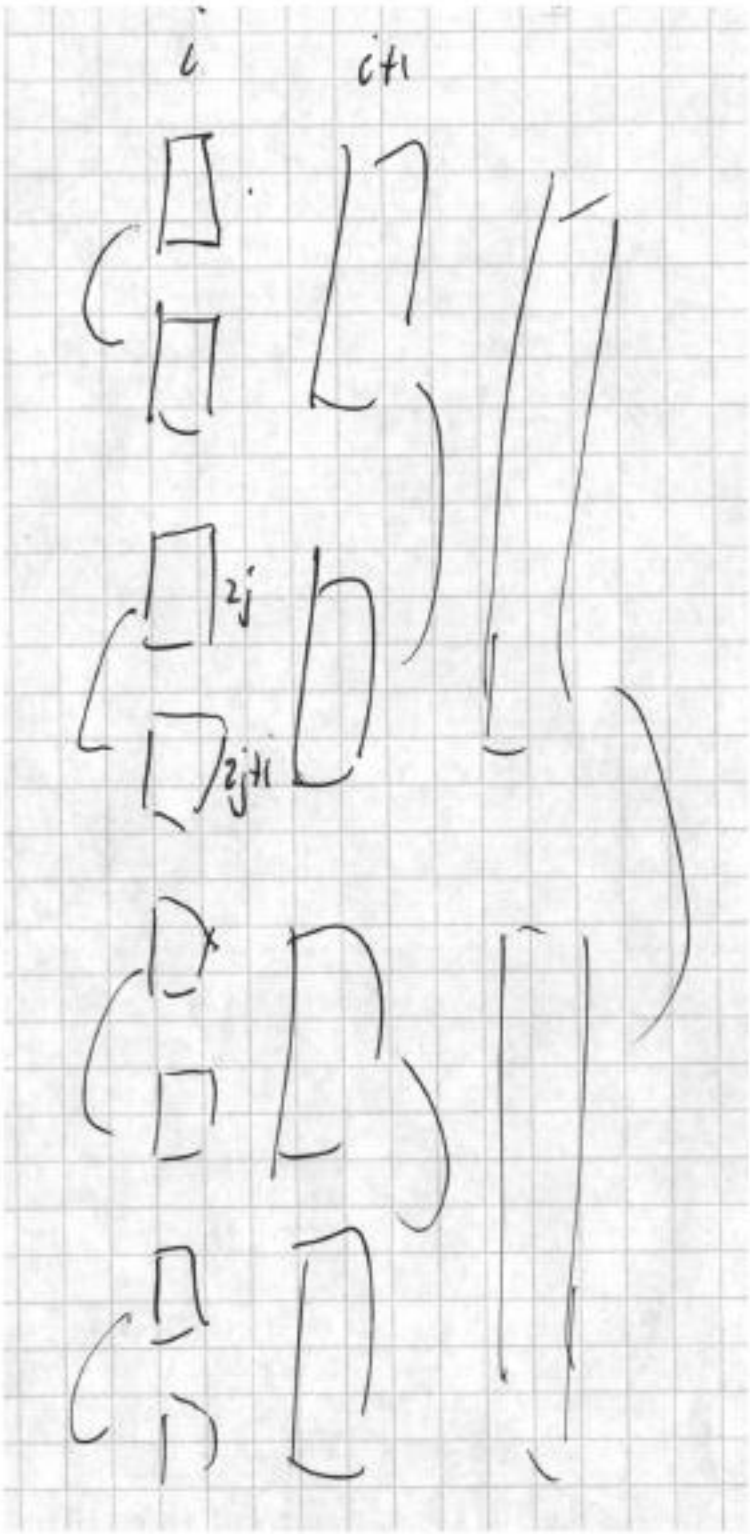


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}|$$

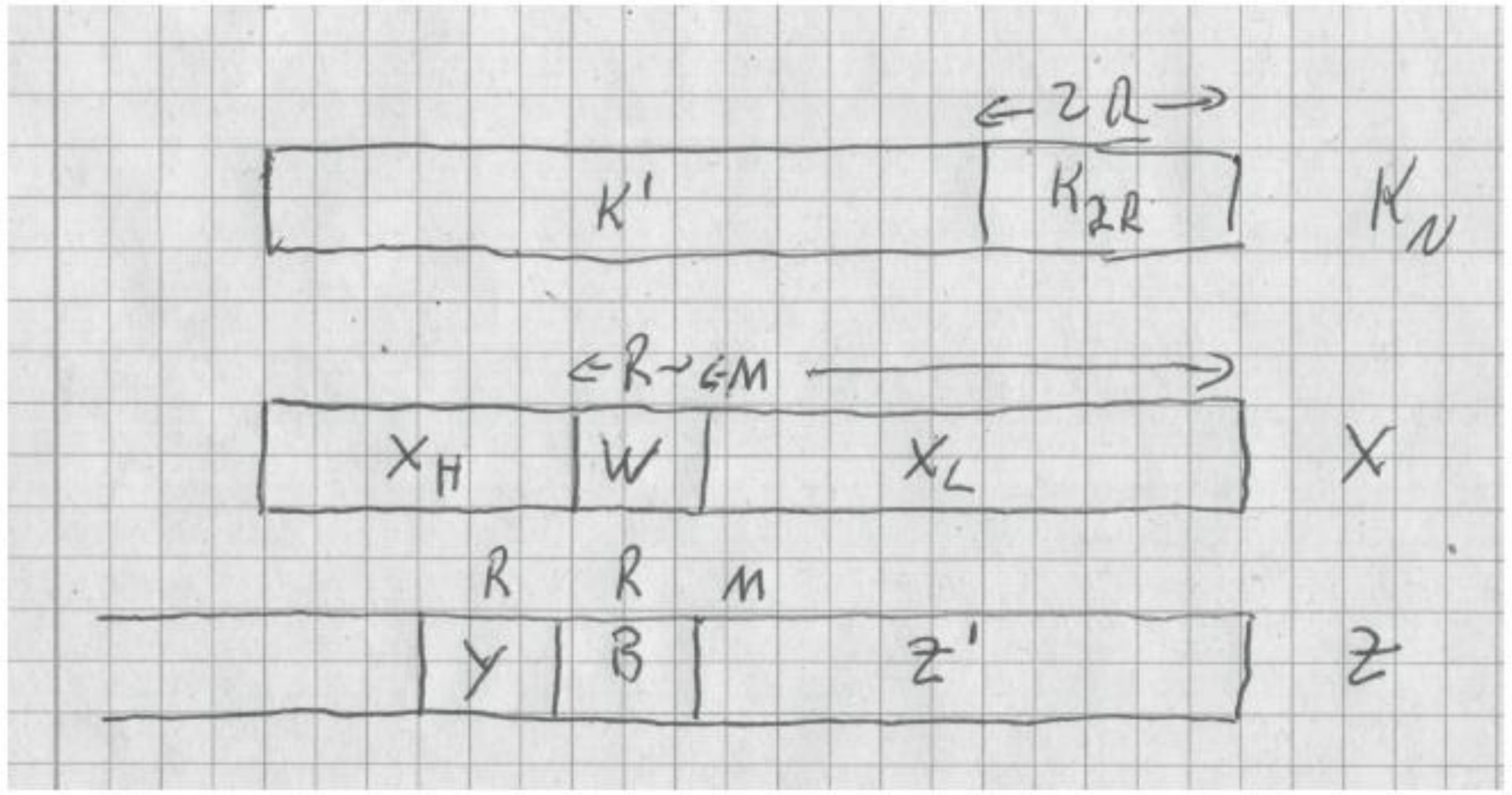
$$|\omega_{i,2j}| = \sum_\tau |\omega^\tau_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N))) \text{ bits}$$

choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1
- substrings W of random strings X are locally almost random, even given the remainder $X_H X_L$ of the string

$$\boxed{K(W \mid X_H X_L) \geq |Y| - O(\log(N)) = 2^i - O(\log(N))}$$



Using $res(i,2j)$ to decode W from $X_H X_L$ with machine D

- input $I\#X_H X_L$
- with extra input $I = bin(|X_H|)' res(i,2j) b$ where $b \in \mathbb{B}$. Length:

$$|I| = |res(i,2j)| + O(\log N)$$

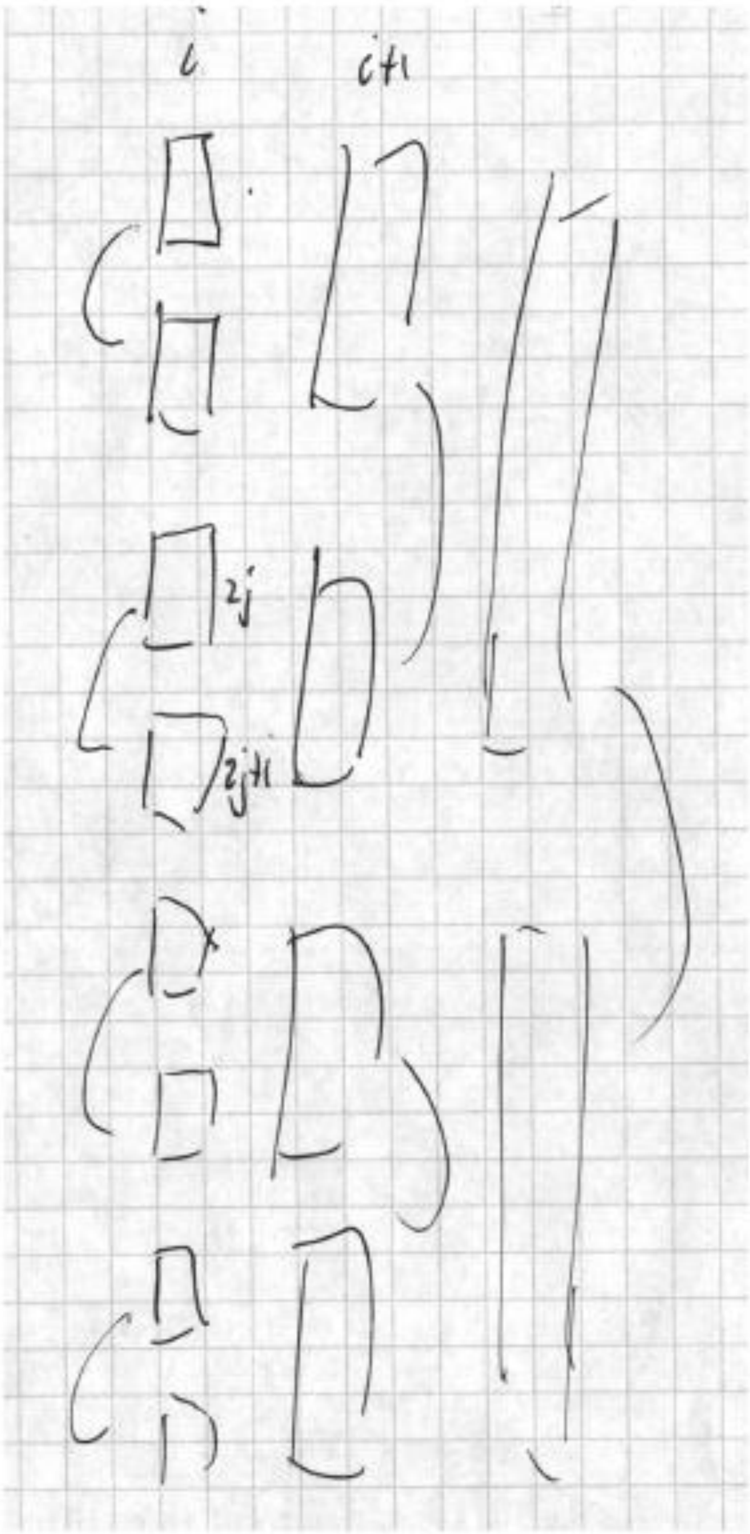


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^{\tau}| = \sum_i \sum_j |E^{\tau}_{i,j}|$$

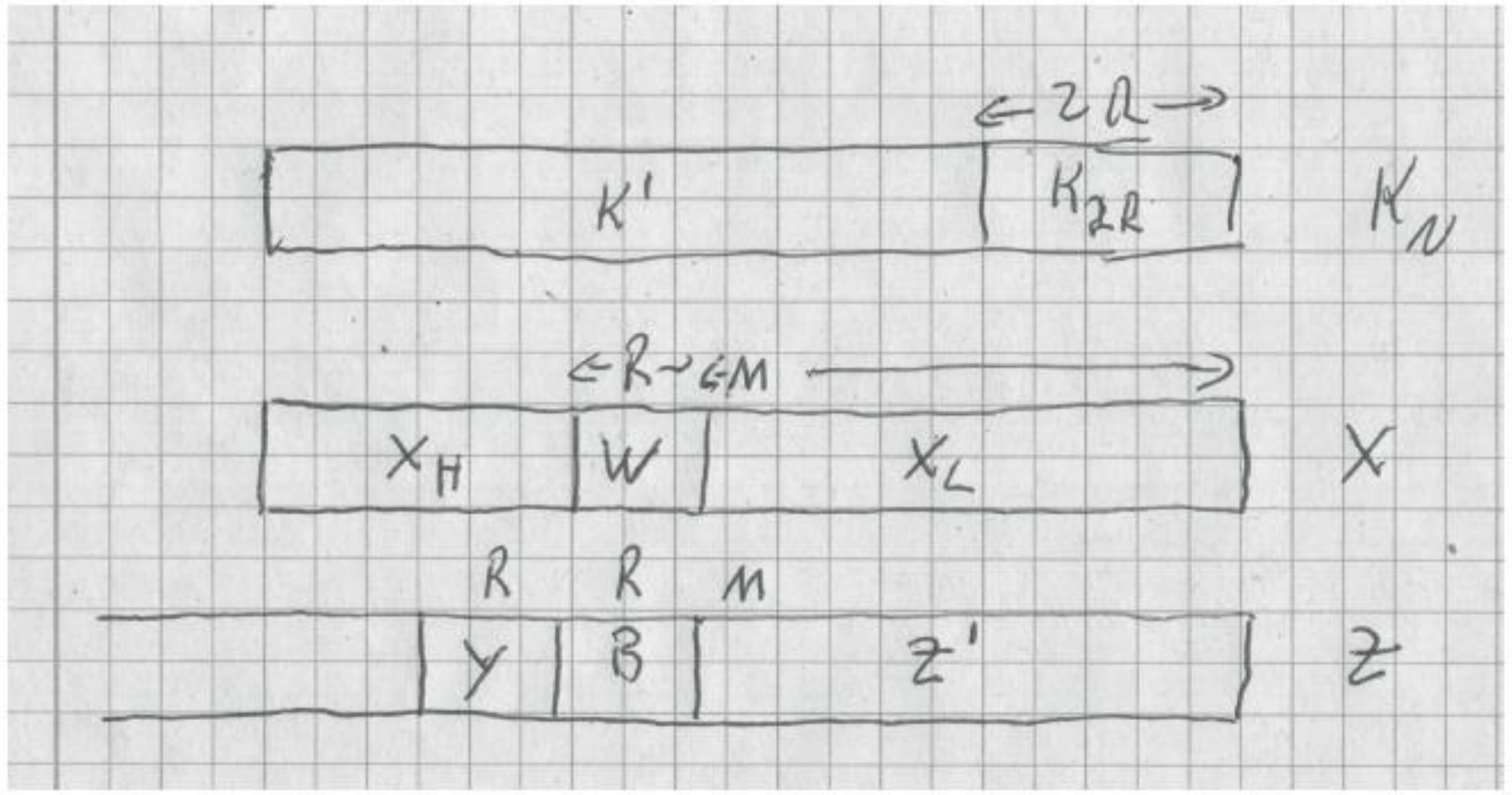
$$|\omega_{i,2j}| = \sum_{\tau} |\omega^{\tau}_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}$$

choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1
- substrings W of random strings X are locally almost random, even given the remainder $X_H X_L$ of the string

$$K(W|\, X_H X_L) \geq |Y| - O(\log(N)) = 2^i - O(\log(N))$$



Using $res(i,2j)$ to decode W from $X_H X_L$ with machine D

- input $I\#X_H X_L$
- with extra input $I = bin(|X_H|)'res(i,2j)b$ where $b \in \mathbb{B}$. Length:

$$|I| = |res(i,2j)| + O(\log N)$$

- machine D runs $Mult$ on input X_L until the start of time interval $T_{i,2j}$ and interrupts in the configuration k at the start of this interval.
- instead of processing input W of this time interval it uses $res(i,2j$ and configuration k to construct configuration k' at the start of time interval $T_{i,2j+1}$

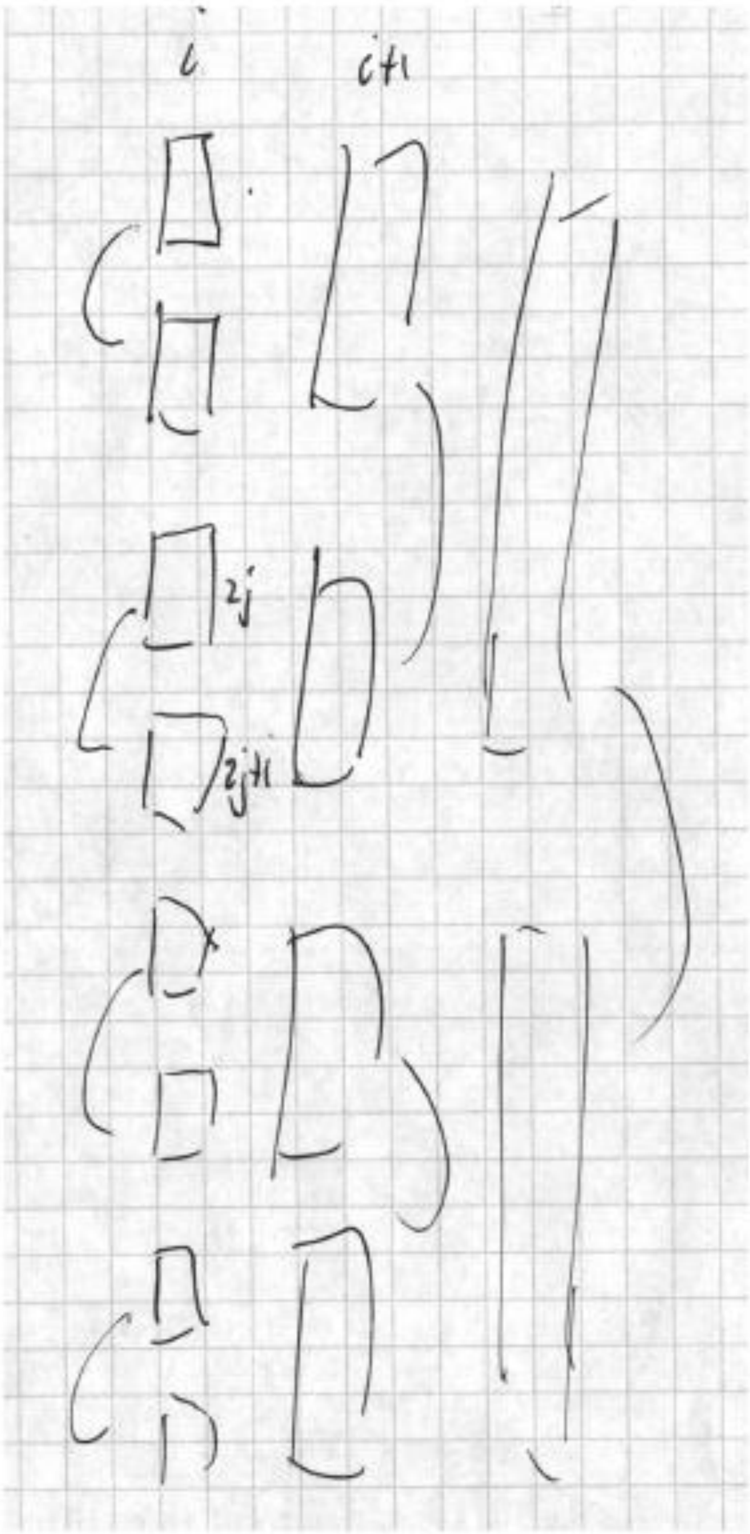


Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$T \geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}|$$

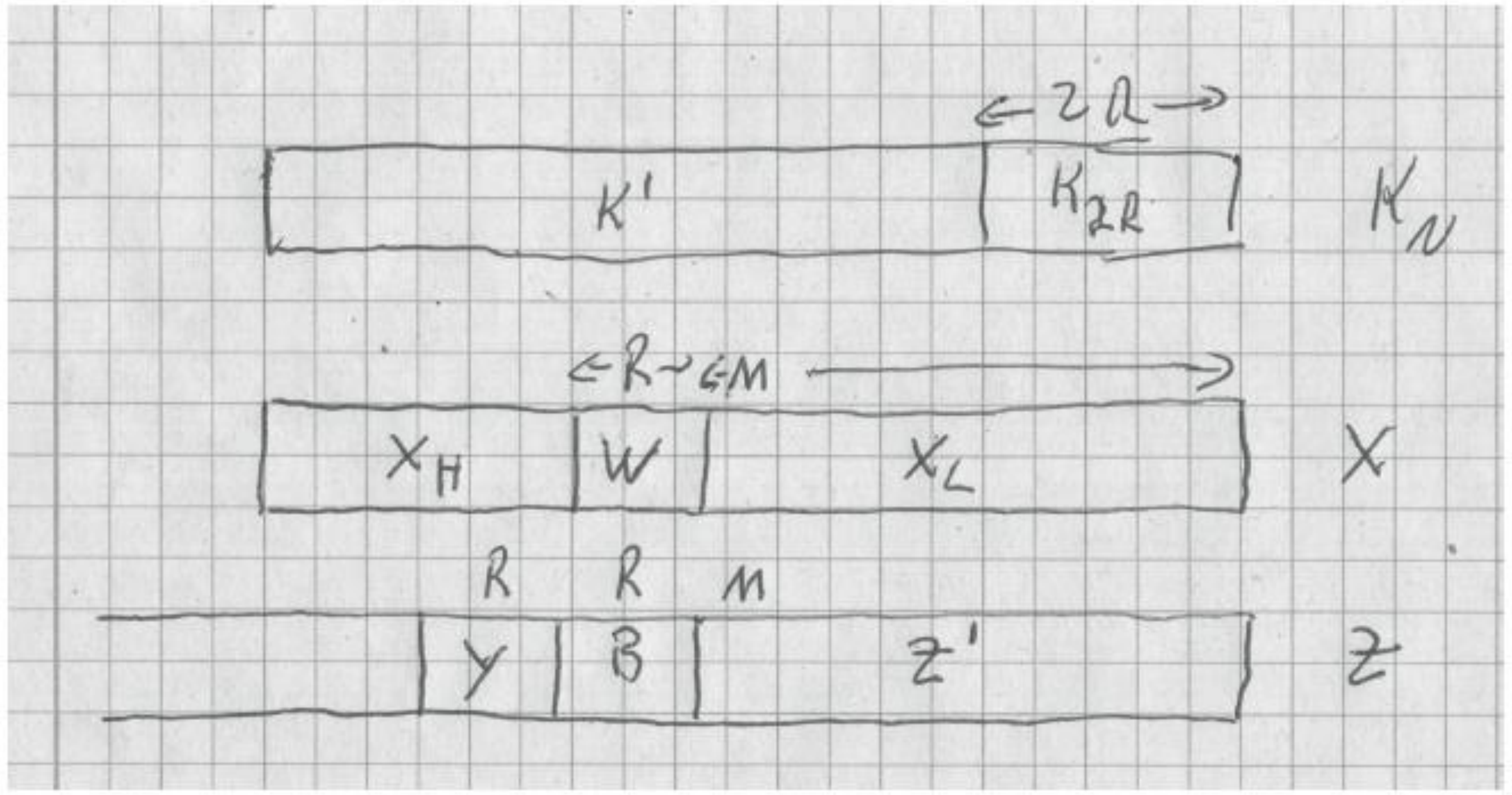
$$|\omega_{i,2j}| = \sum_\tau |\omega^\tau_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}$$

choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1
- substrings W of random strings X are locally almost random, even given the remainder $X_H X_L$ of the string

$$K(W \mid X_H X_L) \geq |Y| - O(\log(N)) = 2^i - O(\log(N))$$



Using $res(i,2j)$ to decode W from $X_H X_L$ with machine D

- input $I\#X_H X_L$
- with extra input $I = bin(|X_H|)'res(i,2j)b$ where $b \in \mathbb{B}$. Length:

$$|I| = |res(i,2j)| + O(\log N)$$

- machine D runs $Mult$ on input X_L until the start of time interval $T_{i,2j}$ and interrupts in the configuration k at the start of this interval.
- instead of processing input W of this time interval it uses $res(i,2j$ and configuration k to construct configuration k' at the start of time interval $T_{i,2j+1}$
- continuing to act like $Mult$ it accesses bits of X_H and produces Y.
- finally it cycles through all strings W and notes which ones produce Y. By lemma 3 there are at most 2 such W. If $b = 0$ it chooses the lexicographically first one, otherwise the second one.

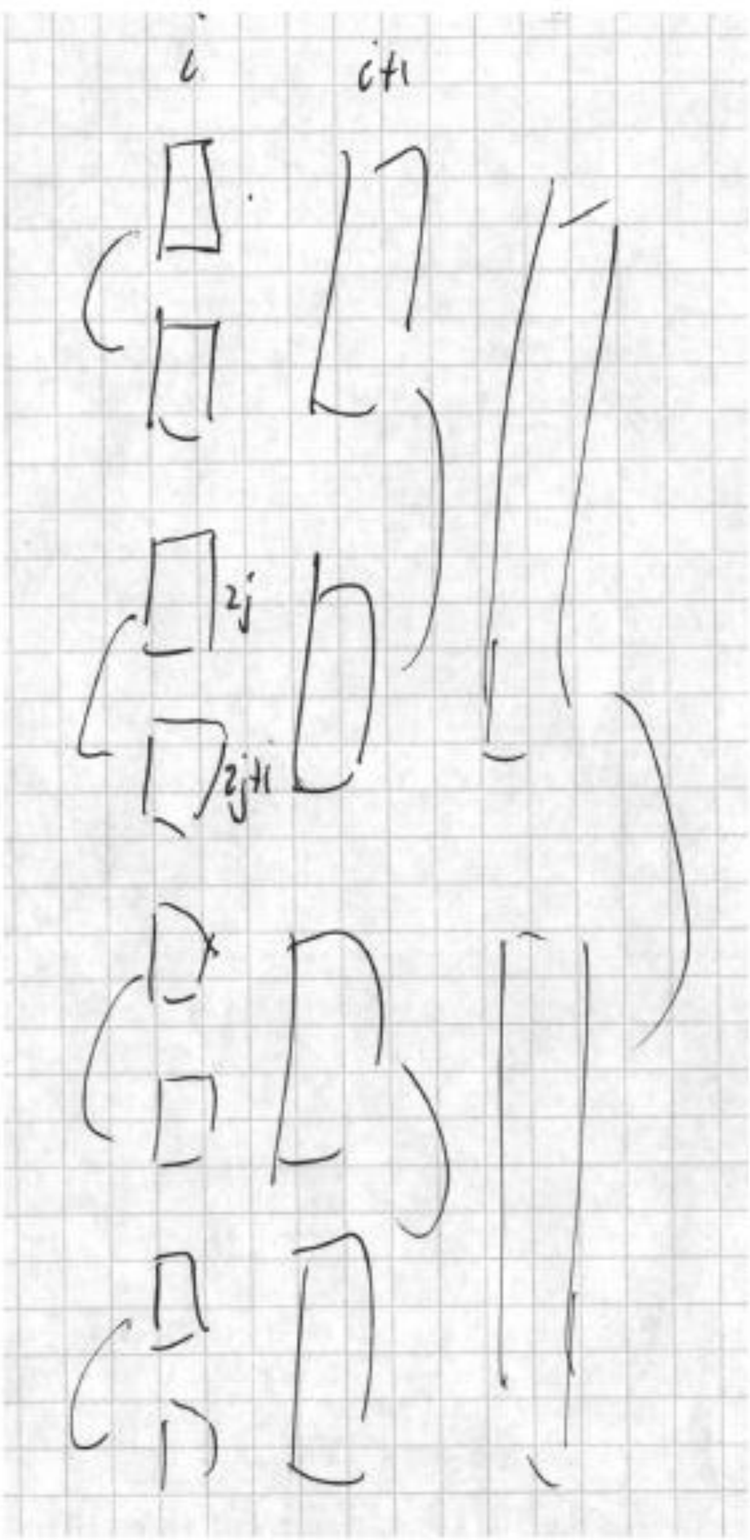


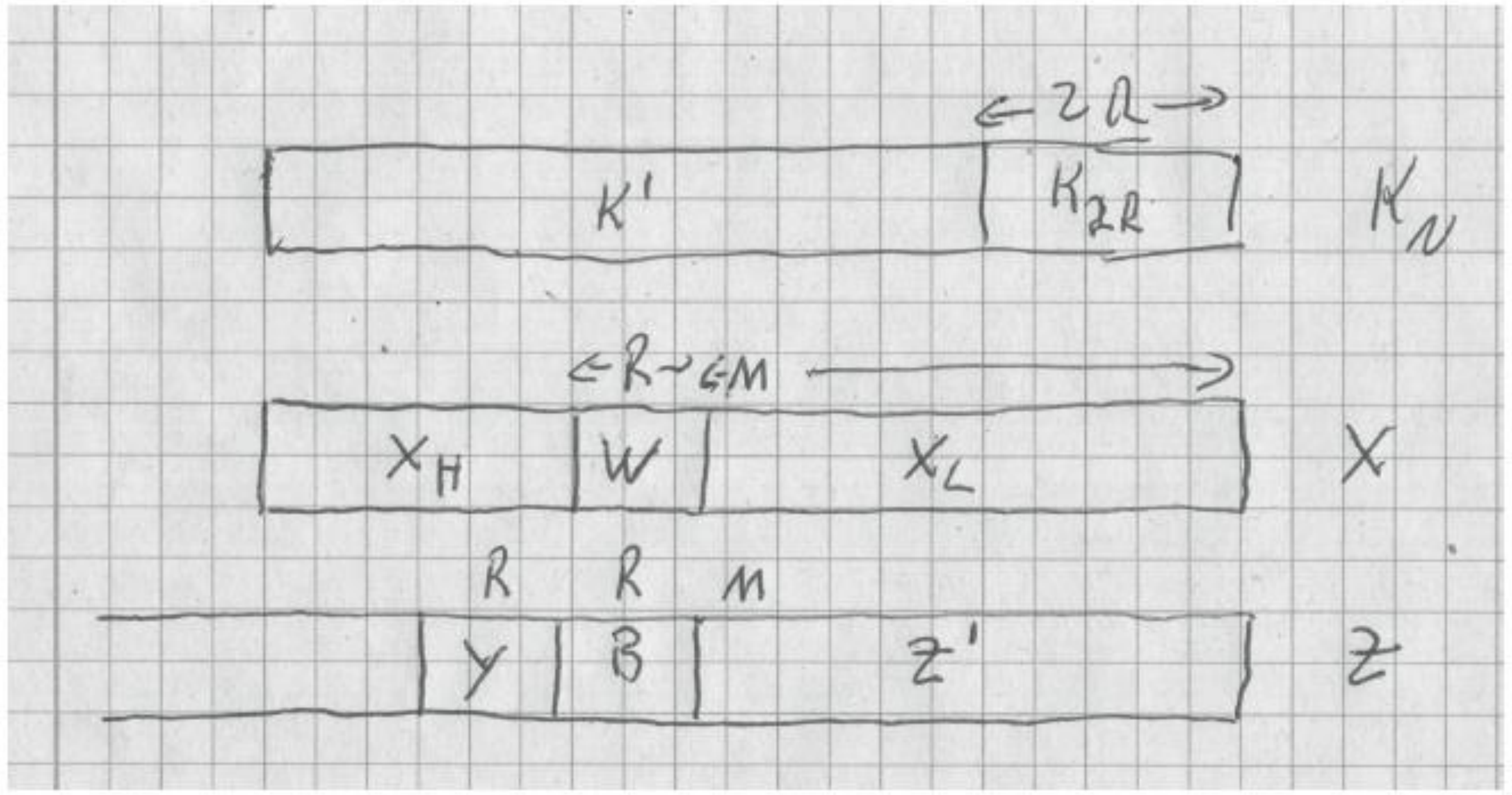
Figure 4: Partitioning edges into set $E_{i,2j}$. Edges in $E_{i,2j}$ go from steps in intervals $T_{i,2j}$ with even indices $2j$ to the next interval $T_{i,2j+1}$

$$\boxed{\begin{aligned} T &\geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}| \\ |\omega_{i,2j}| &= \sum_\tau |\omega^\tau_{i,2j}| \\ |res(i,2j)| &\leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits} \end{aligned}}$$

choosing Kolmogorov random input X

- choose $X \in \mathbb{B}^n$ as a Kolmogorov random string.
- let $R = 2^i$ be the length of strings $W = X_{i,j}$ and $K_{i,j}$ read in time interval $T_{i,2j}$ and of the string Y output during time interval $T_{i,2j+1}$ as shown in figure 1
- substrings W of random strings X are locally almost random, even given the remainder $X_H X_L$ of the string

$$\boxed{K(W|\, X_H X_L) \geq |Y| - O(\log(N)) = 2^i - O(\log(N)}$$



Using $res(i,2j)$ to decode W from $X_H X_L$ with machine D

- input $I\#X_H X_L$
- with extra input $I = bin(|X_H|)' res(i,2j)b$ where $b \in \mathbb{B}$. Length:

$$|I| = |res(i,2j)| + O(\log N)$$

lower bound on time

- bounding size of overlap:

$$\begin{aligned} 2^i - O(\log(N) &\leq K(Y|X_H X_L) \\ &\leq O(1) + |I| \\ &\leq |res(i,2j)| + O(\log N) \\ &\leq c \cdot |\omega_{i,2j}| + O(\log n) \end{aligned}$$

$$\boxed{2^i - C \cdot \log(N) = c \cdot \sum_\tau \omega^\tau_{i,2j}}$$

for some C, N_0 and all $N \geq N_0$

$$\boxed{\begin{array}{l} T \geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}| \\ |\omega_{i,2j}| = \sum_\tau |\omega^\tau_{i,2j}| \\ |res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits} \end{array}}$$

lower bound on time

- bounding size of overlap:

$$\begin{aligned} 2^i - O(\log(N) &\leq K(Y|X_H X_L) \\ &\leq O(1) + |I| \\ &\leq |res(i,2j)| + O(\log N) \\ &\leq c \cdot |\omega_{i,2j}| + O(\log n) \\ \boxed{2^i - C \cdot \log(N) = c \cdot \sum_\tau \omega^\tau_{i,2j}} \end{aligned}$$

for some C, N_0 and all $N \geq N_0$

$$T \geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}|$$

$$|\omega_{i,2j}| = \sum_\tau |\omega^\tau_{i,2j}|$$

$$|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits}$$

lower bound on time

- bounding size of overlap:

$$\begin{aligned}
2^i - O(\log(N) &\leq K(Y|X_H X_L) \\
&\leq O(1) + |I| \\
&\leq |res(i,2j)| + O(\log N) \\
&\leq c \cdot |\omega_{i,2j}| + O(\log n) \\
\boxed{2^i - C \cdot \log(N) = c \cdot \sum_\tau \omega^\tau_{i,2j}}
\end{aligned}$$

  for some C, N_0 and all $N \geq N_0$

- for $i \geq (\log N)/2$ holds for some $N_1 \geq N_0$ and all $N \geq N_1$

$$\begin{aligned}
2^i &\geq \sqrt{N} \\
&\geq 2C \cdot \log N \\
2^i &\geq 2^{i-1} + C \cdot \log N \\
2^i - C \cdot \log N &\geq 2^{i-1}
\end{aligned}$$

- bounding time

$$\begin{aligned}
\sum_{i \geq (\log N)/2} \sum_{2j < N/2^i} 2^{i-1} &\leq \sum_{i \geq (\log N)/2} \sum_j (2^i - C \cdot \log(N)) \\
&\leq c \cdot \sum_\tau \sum_i \sum_j \omega^\tau_{i,2j} \\
&\leq ck \cdot T(N)
\end{aligned}$$

$$\boxed{\begin{aligned} T \geq |E^\tau| &= \sum_i \sum_j |E^\tau_{i,j}| \\ |\omega_{i,2j}| &= \sum_\tau |\omega^\tau_{i,2j}| \\ |res(i,2j)| &\leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits} \end{aligned}}$$

lower bound on time

- bounding size of overlap:

$$\begin{aligned} 2^i - O(\log(N) &\leq K(Y|X_H X_L) \\ &\leq O(1) + |I| \\ &\leq |res(i,2j)| + O(\log N) \\ &\leq c \cdot |\omega_{i,2j}| + O(\log n) \end{aligned}$$

$$\boxed{2^i - C \cdot \log(N) = c \cdot \sum_\tau \omega^\tau_{i,2j}}$$

for some C, N_0 and all $N \geq N_0$

- for $i \geq (\log N)/2$ holds for some $N_1 \geq N_0$ and all $N \geq N_1$

$$\begin{aligned} 2^i &\geq \sqrt{N} \\ &\geq 2C \cdot \log N \\ 2^i &\geq 2^{i-1} + C \cdot \log N \\ 2^i - C \cdot \log N &\geq 2^{i-1} \end{aligned}$$

- bounding time

$$\begin{aligned} \sum_{i \geq (\log N)/2} \sum_{2j < N/2^i} 2^{i-1} &\leq \sum_{i \geq (\log N)/2} \sum_j (2^i - C \cdot \log(N)) \\ &\leq c \cdot \sum_\tau \sum_i \sum_j \omega^\tau_{i,2j} \\ &\leq ck \cdot T(N) \end{aligned}$$

- evaluating left hand side

$$\begin{aligned} \sum_{i \geq (\log N)/2} \sum_{2j < N/2^i} 2^{i-1} &= \sum_{i \geq (\log N)/2} 2^{i-1} \cdot N/2^{i+1} \\ &= \frac{1}{4} \sum_{i \geq (\log N)/2} N \\ &= \frac{1}{8} N \log N \end{aligned}$$

$$\boxed{\begin{aligned} &T \geq |E^\tau| = \sum_i \sum_j |E^\tau_{i,j}| \\ &|\omega_{i,2j}| = \sum_\tau |\omega^\tau_{i,2j}| \\ &|res(i,2j)| \leq c \cdot |\omega_{i,2j}| + O(\log(t(N)) \text{ bits} \end{aligned}}$$

lower bound on time

- bounding size of overlap:

$$\begin{aligned} 2^i - O(\log(N) &\leq K(Y|X_H X_L) \\ &\leq O(1) + |I| \\ &\leq |res(i,2j)| + O(\log N) \\ &\leq c \cdot |\omega_{i,2j}| + O(\log n) \end{aligned}$$

$$\boxed{2^i - C \cdot \log(N) = c \cdot \sum_\tau \omega^\tau_{i,2j}}$$

for some C, N_0 and all $N \geq N_0$

- for $i \geq (\log N)/2$ holds for some $N_1 \geq N_0$ and all $N \geq N_1$

$$\begin{aligned} 2^i &\geq \sqrt{N} \\ &\geq 2C \cdot \log N \\ 2^i &\geq 2^{i-1} + C \cdot \log N \\ 2^i - C \cdot \log N &\geq 2^{i-1} \end{aligned}$$

- bounding time

$$\begin{aligned} \sum_{i \geq (\log N)/2} \sum_{2j < N/2^i} 2^{i-1} &\leq \sum_{i \geq (\log N)/2} \sum_j (2^i - C \cdot \log(N)) \\ &\leq c \cdot \sum_\tau \sum_i \sum_j \omega^\tau_{i,2j} \\ &\leq ck \cdot T(N) \end{aligned}$$

- evaluating left hand side

$$\begin{aligned} \sum_{i \geq (\log N)/2} \sum_{2j < N/2^i} 2^{i-1} &= \sum_{i \geq (\log N)/2} 2^{i-1} \cdot N/2^{i+1} \\ &= \frac{1}{4} \sum_{i \geq (\log N)/2} N \\ &= \frac{1}{8} N \log N \end{aligned}$$

- here it is:

$$\boxed{T(N) \geq \frac{1}{8ck} N \log N}$$