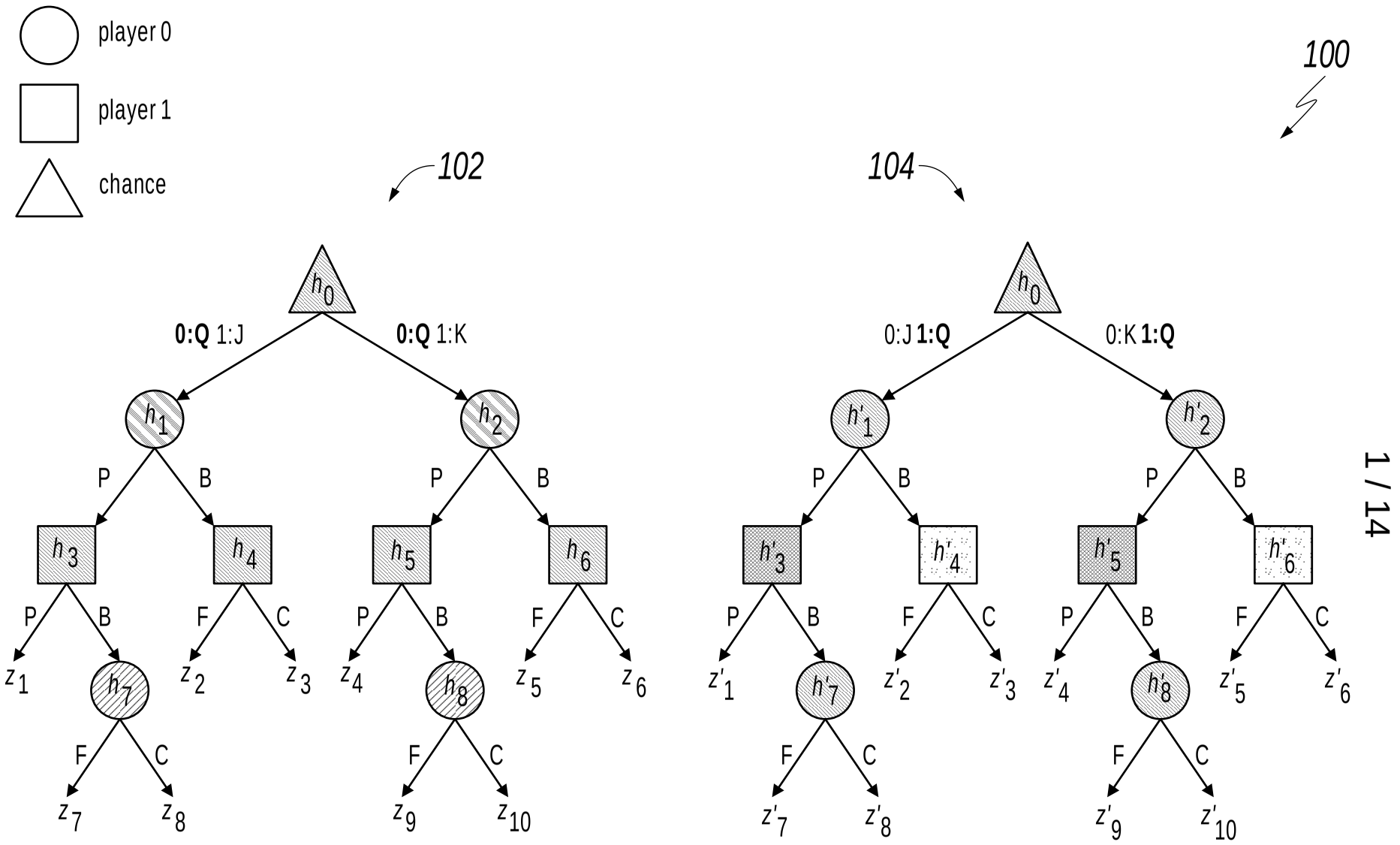

player 0
player 1
chance
100
102
104
h_0
0:Q 1:J
0:Q 1:K
h_1
h_2
P
B
h_3
h_4
h_5
h_6
F
C
z_1
z_2
z_3
z_4
z_5
z_6
h_7
h_8
z_7
z_8
z_9
z_{10}
h'_0
0:J 1:Q
0:K 1:Q
h'_1
h'_2
h'_3
h'_4
h'_5
h'_6
z'_1
z'_2
z'_3
z'_4
z'_5
z'_6
h'_7
h'_8
z'_7
z'_8
z'_9
z'_{10}

FIG. 1

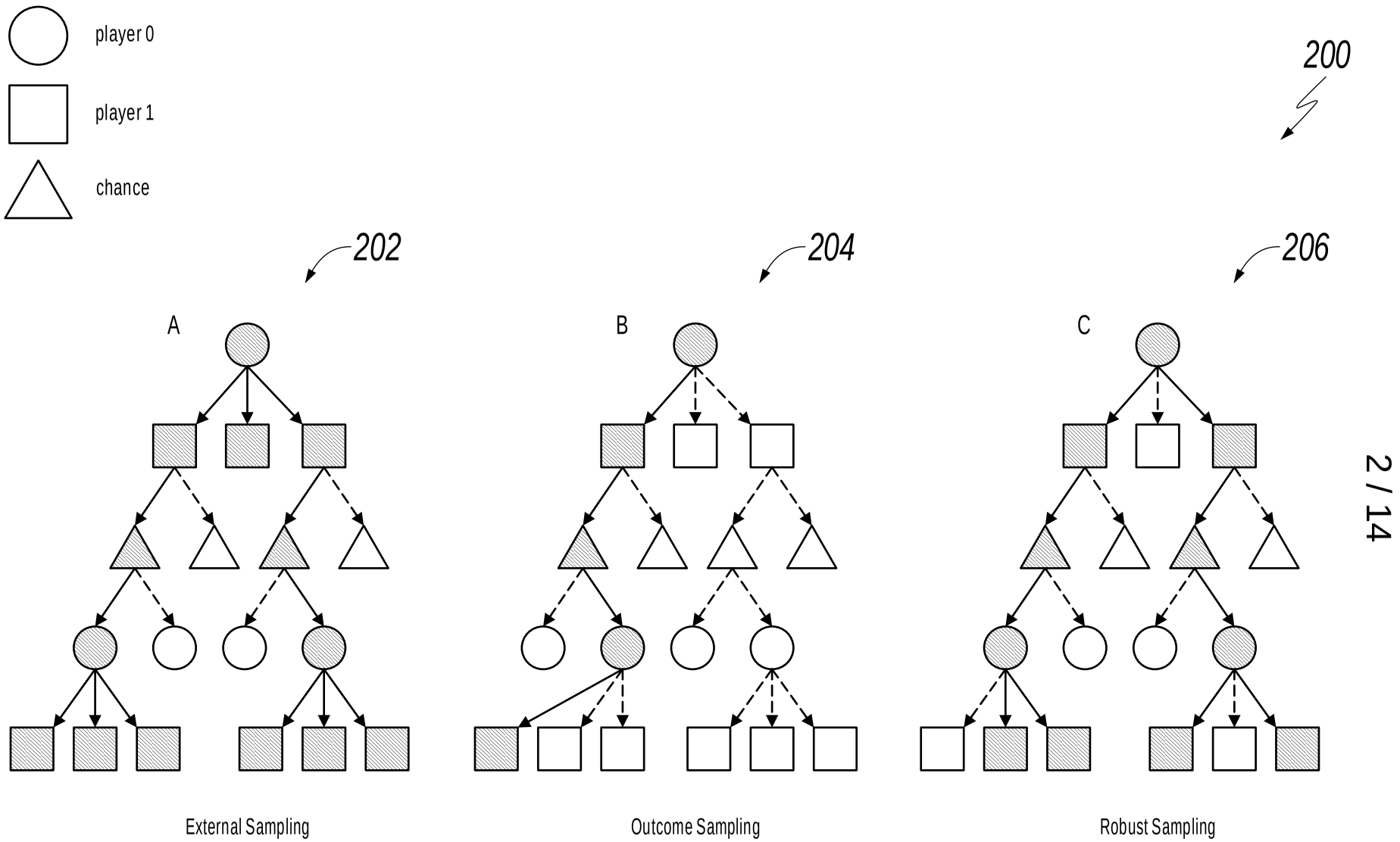

player 0
player 1
chance
200
202
204
206
A
B
C
External Sampling
Outcome Sampling
Robust Sampling

FIG. 2

Algorithm 1: Robust Sampling Monte Carlo Counterfactual Regret Minimization

Function RS-MCCFR-ALG (t) **:**
  for $i = 1$ *to* t **do**
    RS-MCCFR($t, \emptyset, 0, 1, 1$)
    RS-MCCFR($t, \emptyset, 1, 1, 1$)

Function RS-MCCFR ($t, h, i, \pi_i, \pi_i^{rs(k)}$) **:**
  $I_i \leftarrow I_i(h)$ ▷ information set at sate h
  if $h \in Z$ **then**
    return $\frac{u_i(h)}{\pi_i rs(k)}$ ▷ return game payoff
  else if $P(h) = -1$ **then**
    $a \sim \sigma_{-i}(I_i)$ ▷ sample an action from $\sigma_{-i}(h)$
    return RS-MCCFR($t, ha, i, \pi_i, \pi_i^{rs(k)}$)
  else if $P(h) = i$ **then**
    $\sigma_i(i_i) \leftarrow$ CalculateStrategy ($R_i^t(\cdot|I_i), I_i$) ▷ calculate current strategy
    $v_i(h) \leftarrow 0, r_i(\cdot|I_i) \leftarrow \vec{0}, s_i(\cdot|I_i) \leftarrow \vec{0}$ ▷ $r_i(\cdot|I_i)$ *and* $_i(\cdot|I_i)$ are two vectors over $A(I_i)$
    $A^{rs(k)}(I_i) \leftarrow$ sampling k different actions according to $\sigma_i^{rs(k)}$
    For $a \in A^{rs(k)}(I_i)$ **do**
      $v_i(a|h) \leftarrow$ RS-MCCFR($t, ha, i, \pi_i \sigma_i(a|I_i), \pi_i^{rs} \sigma_i^{rs(k)}(a|I_i)$)
      $v_i(h) \leftarrow v_i(h) + v_i(a|h)\sigma_i(a|I_i)$ ▷ update counterfactual value
    For $a \in A^{rs(k)}(I_i)$ **do**
      if $t == 0$ **then**

300

FIG. 3

22 $R_i^t(\cdot|I_i) \leftarrow R_i^{t-1}(\cdot|I_i) + (v_i(a|h) - v_i(h))$ ▷ update cumulative regret

23 $S_i^t(a|I_i) \leftarrow S_i^{t-1}(a|I_i) + \pi_i^{\sigma}(I_i)\,\sigma_i(a|I_i)$ ▷ update average strategy numerator

24 **else**

25 $R_i^t(\cdot|I_i) \leftarrow (v_i(a|h) - v_i(h))$ ▷ update cumulative regret

26 $S_i^t(a|I_i) \leftarrow \pi_i^{\sigma}(I_i)\sigma_i(a|I_i)$ ▷ update average strategy numerator

27 **else**

28 $\sigma_{-i}(I_i) \leftarrow$ CalculateStrategy $(R_{-i}^t(\cdot|I_i), I_i)$ ▷ calculate current strategy

29 $a \sim \sigma_{-i}(I_i)$ ▷ sample an action from $\sigma_{-i}(I_i)$

30 return RS-MCCFR$(t, ha, i, \pi_i, \pi_i^{rs(k)})$

31

32 **Function** CalculateStrategy $(R_i(\cdot|I_i), I_i)$ **:**

33 $sum \leftarrow \Sigma_{a \in A(I_i)} \max(R_i(a|I_i), 0)$

34 **For** $a \in A(I_i)$ **do**

35 $\sigma_i(a|I_i) = \frac{\max(R_i(a|I_i), 0)}{sum}$ **if** sum > 0 **else** $\frac{1}{|A(I_i)|}$

36 return $\sigma_i(I_i)$

37

300

FIG. 3 (continued)

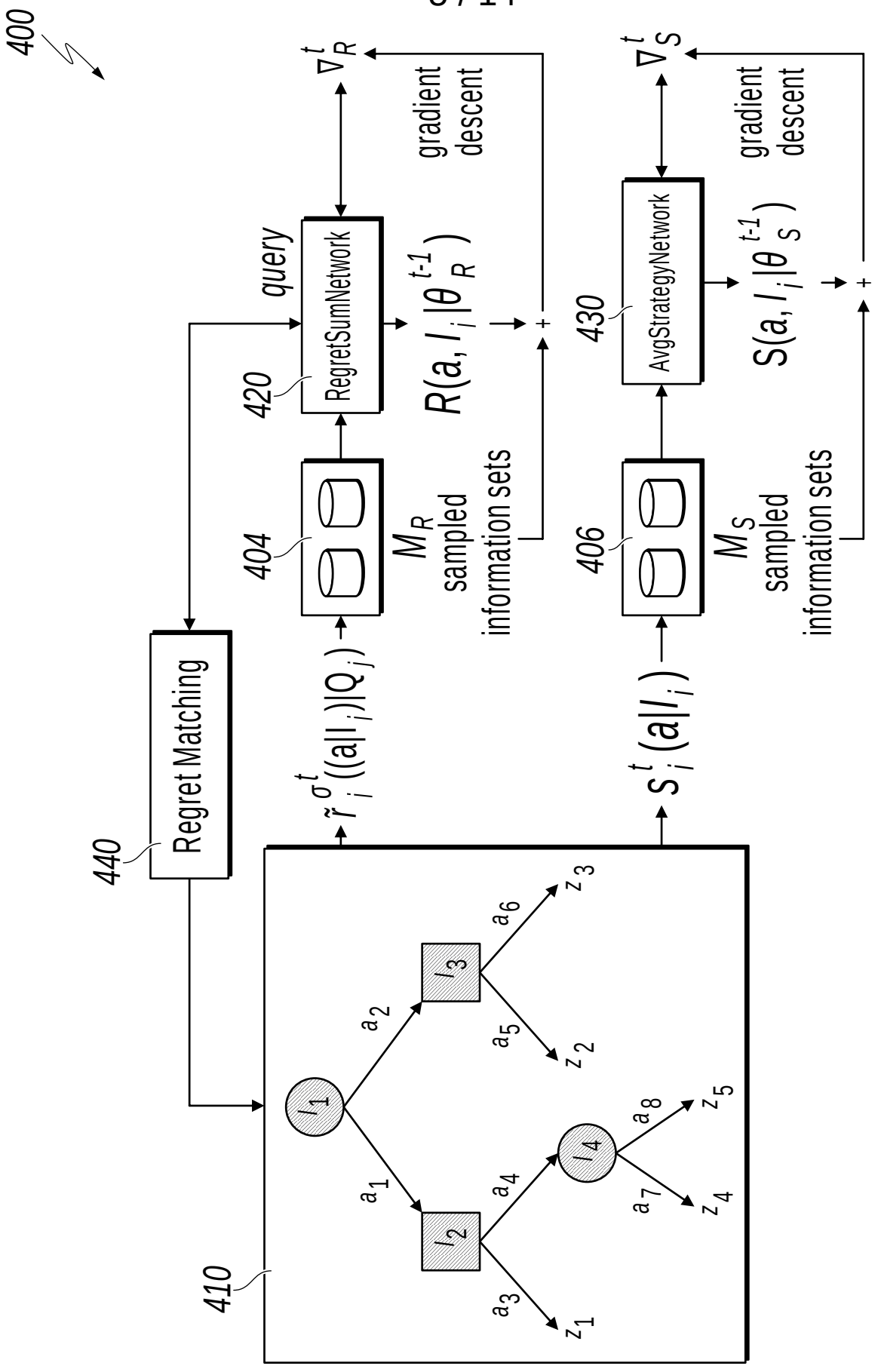

400
410
440
Regret Matching
$\tilde{r}_i^{\sigma^t}((a|I_i)|Q_j)$
404
M_R
sampled
information sets
420
RegretSumNetwork
query
$R(a, I_i|\theta_R^{t-1})$
gradient
descent
∇_R^t
$s_i^t(a|I_i)$
406
M_S
sampled
information sets
430
AvgStrategyNetwork
$S(a, I_i|\theta_S^{t-1})$
gradient
descent
∇_S^t
I_1
I_2
I_3
I_4
a_1
a_2
a_3
a_4
a_5
a_6
a_7
a_8
z_1
z_2
z_3
z_4
z_5

FIG. 4



Algorithm 1: Counterfactual Regret Minimization with Two Deep Neural Networks

Function Agent (T, b) **:**
  For $t = 1$ *to* T **do**
    if $t = 1$ ***and*** *using warm starting* **then**
      initialize θ_R^t and θ_S^t from an existing checkpoint
      $t \leftarrow t + 1$ ▷ skip cold starting
    else
      initialize θ_R^t and θ_S^t randomly
    $M_R^t, M_S^t \leftarrow$ sampling methods ▷ such as Algorithm3
    sum aggregate value in M_R by information set ▷ according to the **Lemma 5** and Equation 13
    remove duplicated records in M_S
    if *use reservoir method* **then**
      $M_R \leftarrow M_R^t$, $M_S \leftarrow M_S^t$
    $\theta_R^t \leftarrow$ NerualAgent($R(\cdot|\theta_R^{t-1})$, M_R^t, θ_R^{t-1}, β_R^*) ▷ update θ_R^t using Algorithm2
    $\theta_S^t \leftarrow$ NerualAgent($S(\cdot|\theta_S^{t-1})$, M_S^t, θ_S^{t-1}, β_S^*) ▷ update θ_S^t using Algorithm2
  return θ_R^t, θ_S^t

FIG. 5

600

Algorithm 2: Optimization of Deep Neural Network

Function NeuralAgent ($f(\cdot|\theta^{t-1}), M, \theta^{t-1}, \beta^*$)
  initialize optimizer, scheduler ▷ gradient descent optimizer and learning rate scheduler
  $\theta^T \leftarrow \theta^{T-1}, l_{best} \leftarrow \infty, t_{best} \leftarrow 0$ ▷ warm starting from the checkpoint of the last iteration
  For $t = 1$ to β_{epoch} **do**
    $loss \leftarrow []$ ▷ initialize *loss* as an empty list
    For each training *epoch* **do**
      $\{x^{(i)}, y^{(i)}\}_{i=1}^{m} \sim M$ ▷ sampling a mini-batch from M
      $batch_loss \leftarrow \frac{1}{m} \Sigma_{i=1}^{m} (f(x^{(i)}|\theta^{T-1}) + y^{(i)} - f(x^{(i)}|\theta^T))^2$
      back propagation batch_loss with learning rate *lr*
      clip gradient of θ^T to $[-\epsilon, \epsilon]^d$ ▷ d is the dimension of θ^T
      optimizer(*batch_loss*)
      loss.append(*batch_loss*)
    $lr \leftarrow sheduler(lr)$ ▷ reduce learning rate adaptively when loss has stopped improving
    if $avg(loss) < \beta_{loss}$ **then**
      $\theta_{best}^T \leftarrow \theta^T$, early stopping ▷ if loss is small enough, using early stopping mechanism
    else if $avg(loss) < l_{best}$ **then**
      $l_{best} = avg(loss), t_{best} \leftarrow t, \theta_{best}^T \leftarrow \theta^T$
    if $t - t_{best} > \beta_{re}$ **then**
      $lr \leftarrow \beta_{lr}$ ▷ reset learning rate to escape from potential saddle point or local minima
  return θ^T

FIG. 6

Algorithm 3: Mini-Batch RS-MCCFR with Double Neural Networks

Function Mini-Batch-MCCFR-NN(t) :
  $M_R^t \leftarrow \emptyset, M_S^t \leftarrow \emptyset$
  For all $i = 1$ *to* b **do in parallel then**
    MCCFR-NN($t, \emptyset, 0, 1, 1$)
    MCCFR-NN($t, \emptyset, 1, 1, 1$)
  return M_R^t, M_S^t

Function MCCFR-NN($t, h, i, \pi_i, \pi_i^{rs(k)}$):
  $I_i \leftarrow I_i(h)$ ▷ information set at sate h
  if $h \in Z$ **then**
    return $\frac{u_i(h)}{\pi_i^{rs(k)}}$ ▷ return game payoff
  else if $P(h) = -1$ **then**
    $a \sim \sigma_{-i}(I_i)$ ▷ sample an action from $\sigma_{-i}(h)$
    return MCCFR-NN($t, ha, i, \pi_i, \pi_i^{rs(k)}$)
  else if $P(h) = i$ **then**
    $\hat{R}_i(\cdot|I_i) \leftarrow R(\cdot, I_i|\theta_R^t)$ **if** $t > 1$ **else** $\vec{0}$ ▷ inference the vector of cumulative regret $\forall a \in A(I_i)$
    $\sigma_i(I_i) \leftarrow$ CalculateStrategy $(\hat{R}_i(\cdot|I_i), I_i)$ ▷ calculate current strategy
    $v_i(h) \leftarrow 0, r_i(\cdot|I_i) \leftarrow \vec{0}, s_i(\cdot|I_i) \leftarrow \vec{0}$ ▷ $r_i(\cdot|I_i)$ *and* $s_i(\cdot|I_i)$ are two vectors over $A(I_i)$
    $A^{rs(k)}(I_i) \leftarrow$ sampling k different actions according to $\sigma_i^{rs(k)}$
    For $a \in A^{rs(k)}(I_i)$ **do**
      $v_i(a|h) \leftarrow$ MCCFR-NN($t, ha, i, \pi_i \sigma_i(a|I_i), \pi_i^{rs} \sigma_i^{rs(k)}(a|I_i)$)
      $v_i(h) \leftarrow v_i(h) + v_i(a|h)\sigma_i(a|I_i)$ ▷ update counterfactual value

FIG. 7

700

```
23          For a ∈ A^{rs(k)}(I_i) do
24              r_i(a|I_i) ← v_i(a|h) - v_i(h)                    ▷ update cumulative regret
25              s_i(a|I_i) ← π_i^σ(I_i) σ_i(a|I_i)                ▷ update average strategy numerator
26          Store updated cumulative regret tuple (I_i, r_i(·|I_i)) in M_R
27          Store updated current strategy dictionary (I_i, s_i(·|I_i)) in M_S
28          return v_i(h)
29      else
30          R̂_i(·|I_i) ← R(·, I_i|θ_R^t) if t > 1 else 0⃗        ▷ inference cumulative regret
31          σ_i(I_i) ← CalculateStrategy(R̂_i(·|I_i), I_i)       ▷ calculate current strategy
32          a ~ σ_i(I_i)                                        ▷ sample an action from σ_i(I_i)
33          return MCCFR-NN(t, ha, i, π_i, π_i^{rs(k)})
34
35  Function CalculateStrategy(R_i(·|I_i), I_i):
36      sum ← Σ_{a∈A(I_i)} max(R_i(a|I_i), 0)
37      For a ∈ A(I_i) do
38          σ_i(a|I_i) = max(R_i(a|I_i), 0) / sum  if sum > 0 else 1 / |A(I_i)|
39      return σ_i(I_i)
40
```

700

FIG. 7(continued)

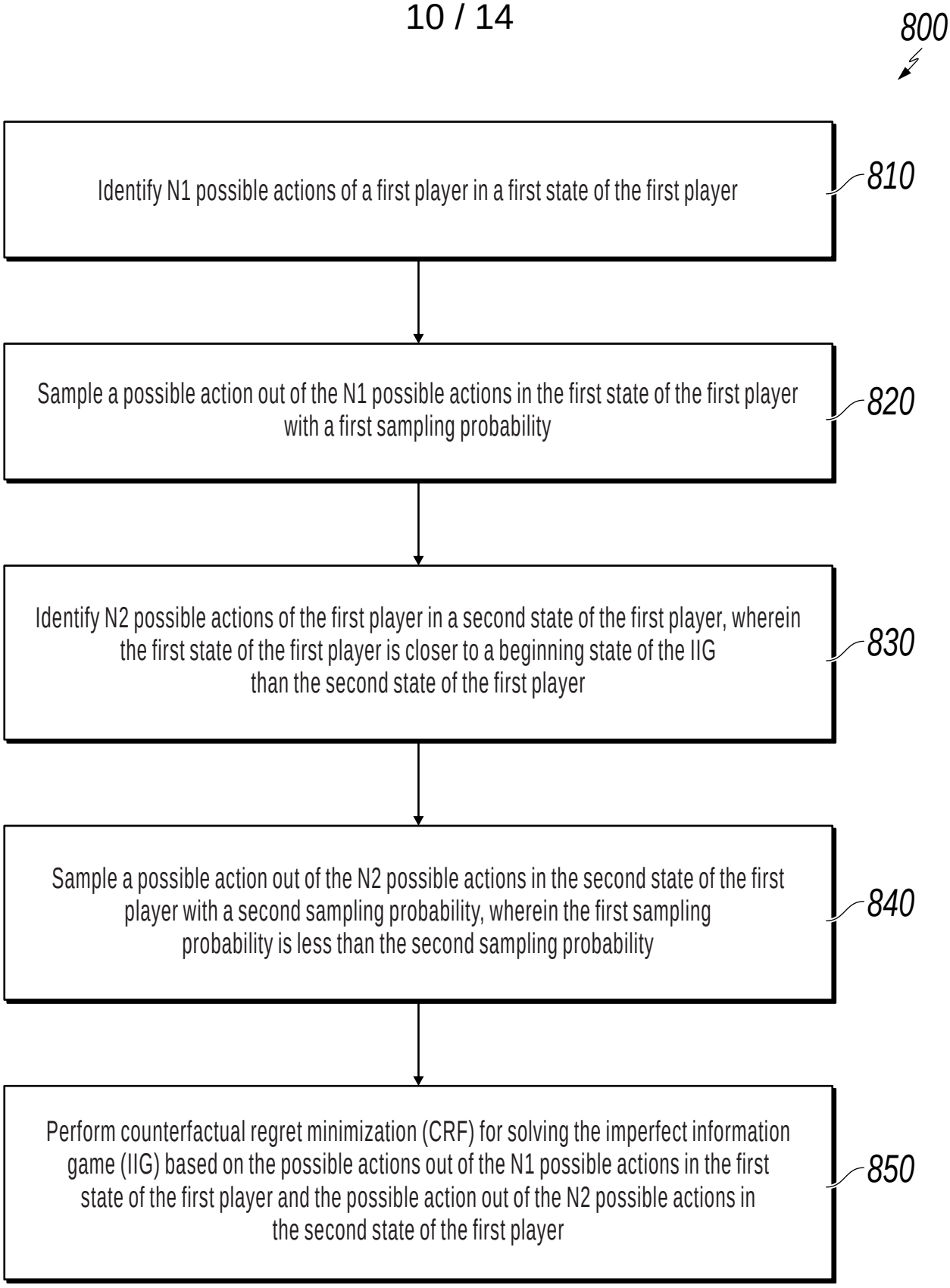

800
Identify N1 possible actions of a first player in a first state of the first player
810
Sample a possible action out of the N1 possible actions in the first state of the first player with a first sampling probability
820
Identify N2 possible actions of the first player in a second state of the first player, wherein the first state of the first player is closer to a beginning state of the IIG than the second state of the first player
830
Sample a possible action out of the N2 possible actions in the second state of the first player with a second sampling probability, wherein the first sampling probability is less than the second sampling probability
840
Perform counterfactual regret minimization (CRF) for solving the imperfect information game (IIG) based on the possible actions out of the N1 possible actions in the first state of the first player and the possible action out of the N2 possible actions in the second state of the first player
850

FIG. 8

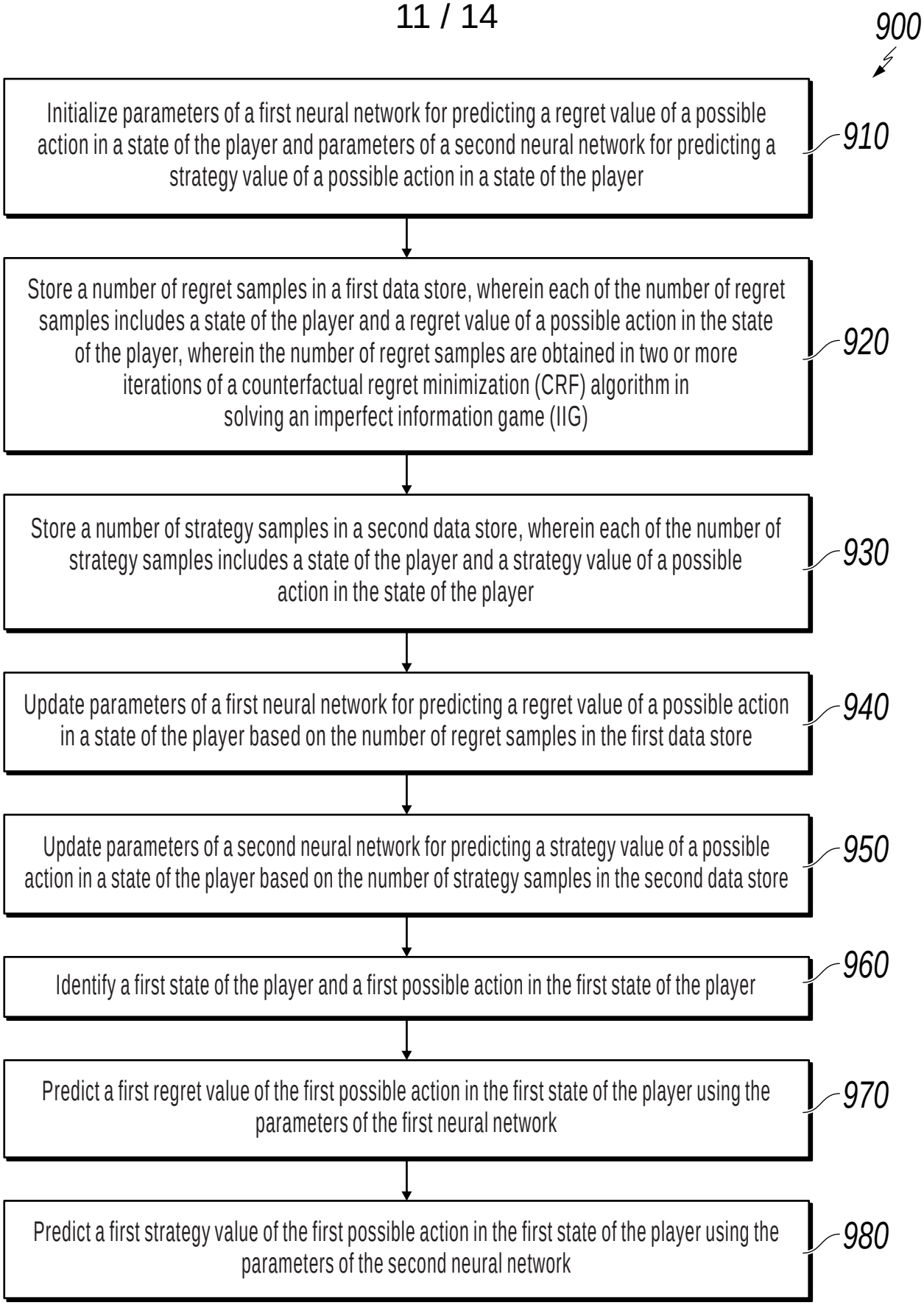

900
Initialize parameters of a first neural network for predicting a regret value of a possible action in a state of the player and parameters of a second neural network for predicting a strategy value of a possible action in a state of the player
910
Store a number of regret samples in a first data store, wherein each of the number of regret samples includes a state of the player and a regret value of a possible action in the state of the player, wherein the number of regret samples are obtained in two or more iterations of a counterfactual regret minimization (CRF) algorithm in solving an imperfect information game (IIG)
920
Store a number of strategy samples in a second data store, wherein each of the number of strategy samples includes a state of the player and a strategy value of a possible action in the state of the player
930
Update parameters of a first neural network for predicting a regret value of a possible action in a state of the player based on the number of regret samples in the first data store
940
Update parameters of a second neural network for predicting a strategy value of a possible action in a state of the player based on the number of strategy samples in the second data store
950
Identify a first state of the player and a first possible action in the first state of the player
960
Predict a first regret value of the first possible action in the first state of the player using the parameters of the first neural network
970
Predict a first strategy value of the first possible action in the first state of the player using the parameters of the second neural network
980

FIG. 9

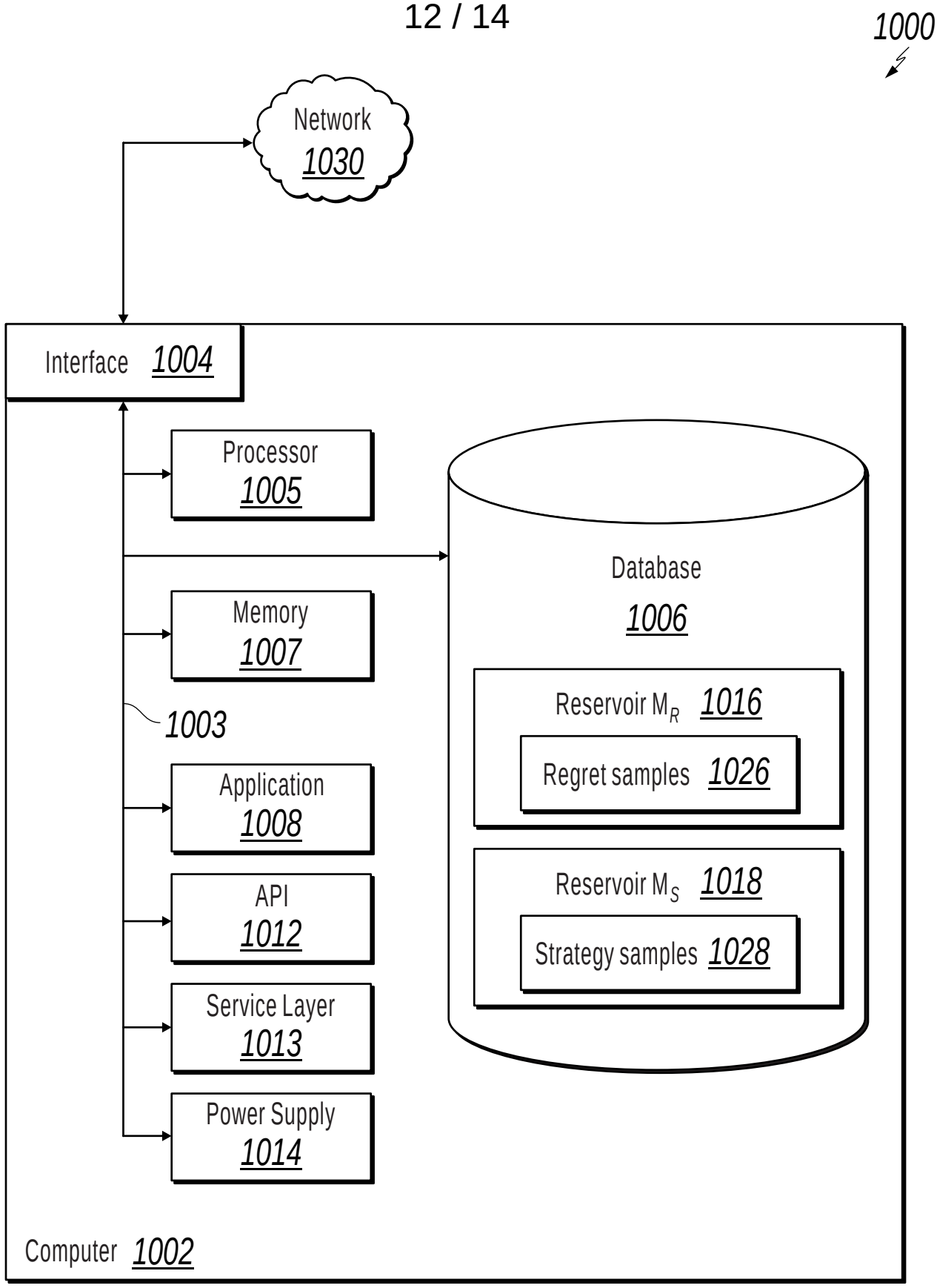

1000
Network
1030
Interface 1004
Processor
1005
Memory
1007
1003
Application
1008
API
1012
Service Layer
1013
Power Supply
1014
Database
1006
Reservoir M_R 1016
Regret samples 1026
Reservoir M_S 1018
Strategy samples 1028
Computer 1002

FIG. 10

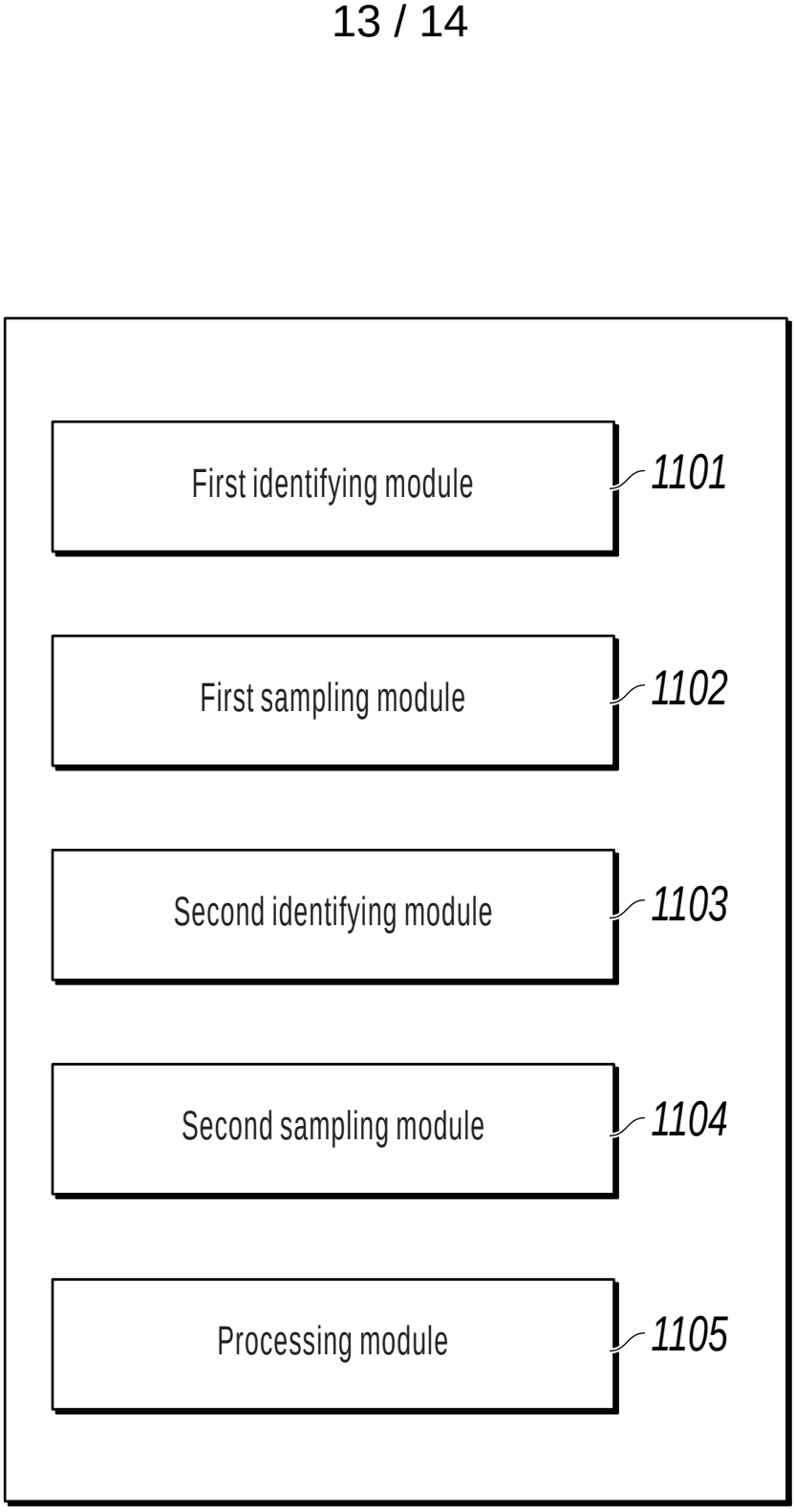

1100
First identifying module
1101
First sampling module
1102
Second identifying module
1103
Second sampling module
1104
Processing module
1105

FIG. 11

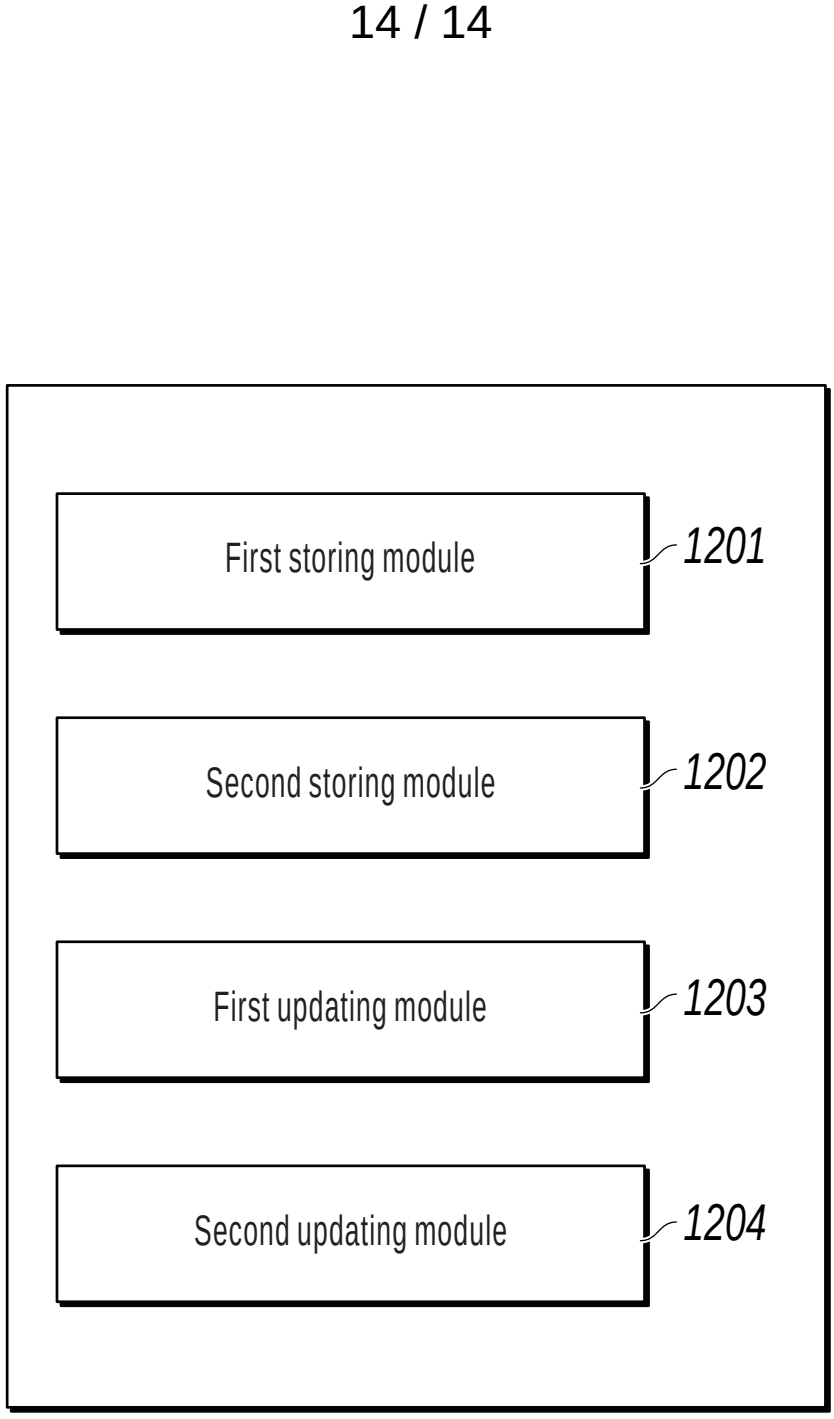

1200
First storing module
1201
Second storing module
1202
First updating module
1203
Second updating module
1204

FIG. 12