

TOKEN MANAGEMENT AND INFORMATION SENDING METHODS AND APPARATUSES, ELECTRONIC DEVICE, AND STORAGE MEDIUM

TECHNICAL FIELD

[0001] The present disclosure relates to the technology field of network communication, and in particular to a token management method and apparatus, an information sending method and apparatus, an electronic device, and a non-transitory computer readable storage medium storing computer instructions.

BACKGROUND

[0002] In a network transmission scenario, there is often a need for rate limiting and shaping of traffic. Main algorithms used for traffic rate limiting and traffic shaping are the token bucket algorithm and the leaky bucket algorithm. The basic principle thereof is the producer-consumer model. A producer produces tokens at a constant rate and uses a token bucket to store the produced tokens, while a consumer randomly arrives to consume the tokens in the token bucket. When there is no token in the token bucket, the consumer cannot consume, thereby achieving the effect of rate limiting and shaping.

[0003] In a high-speed network transmission scenario, the network traffic is quite large, and even a single network device cannot meet the traffic demand, and thus there is a multi-machine load application scenario. However, traditional token allocation algorithms cannot achieve multi-machine load balancing, resulting in low efficiency in both token scheduling and information sending.

SUMMARY

[0004] The present disclosure provides at least a token management method and apparatus, an information sending method and apparatus, an electronic device, and a non-transitory computer readable storage medium storing computer instructions, to solve at least one of the above technical defects.

[0005] According to an aspect of the present disclosure, a token management method is provided. The method is applied to a first token proxy module and includes:

[0006] determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module; and

[0007] acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information; wherein the preset queue includes multiple nodes, one of the multiple nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period.

[0008] In a possible implementation manner, the token management method further includes:

[0009] in response to receiving the number of surplus tokens reported by the at least one second token proxy module, acquiring a total number of surplus tokens from a node corresponding to a current statistical period, updating the total number of surplus tokens corresponding to the current statistical period according to the at least one received number of surplus tokens, and using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period; wherein respective nodes of the preset queue store respective total numbers of surplus tokens in a chronological order.

[0010] In a possible implementation manner, acquiring the total number of surplus tokens from the node corresponding to the current statistical period, includes:

[0011] acquiring a first position of a first pointer in the preset queue; wherein the first pointer is used for pointing to the node, corresponding to the current statistical period, in the preset queue; and

[0012] acquiring, according to the first position, the total number of surplus tokens from the node corresponding to the current statistical period.

[0013] Using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period, includes:

[0014] emptying the node corresponding to the current statistical period according to the first position, and storing the updated total number of surplus tokens into the node corresponding to the current statistical period; and

[0015] at the end of the current statistical period, pointing, according to a preset direction, the first pointer to a next node of the node corresponding to the current statistical period; wherein the preset queue includes a circular queue.

[0016] In a possible implementation manner, acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, includes:

[0017] comparing a total number of surplus tokens stored in a token node of the preset queue with the number of lacked tokens; wherein the token node is a non-empty node with an earliest time in the preset queue;

[0018] in a case where the total number of surplus tokens stored in the token node is less than the number of lacked tokens, extracting the surplus tokens from the token node, and updating the number of lacked tokens according to the number of the extracted surplus tokens; and

[0019] taking a next node of the token node in the preset queue as a new token node, and returning the step of comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens, until a total number of the extracted surplus tokens is equal to the number of lacked tokens carried in the token demand information.

[0020] In a possible implementation manner, acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, further includes:

[0021] in a case where the total number of surplus tokens stored in the token node is larger than the number of lacked tokens, extracting surplus tokens of a same number as the number of lacked tokens from the token node, and updating the total number of surplus tokens stored in the token node; and/or

[0022] in a case where the total number of surplus tokens stored in the token node is equal to the number of lacked tokens, extracting the surplus tokens from the token node, emptying the token node, and taking the next node of the token node in the preset queue as the new token node.

[0023] In a possible implementation manner, comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens, includes:

[0024] acquiring a second position of a second pointer in the preset queue; wherein the second pointer is used for pointing to the token node in the preset queue;

[0025] acquiring, according to the second position, the total number of surplus tokens stored in the token node from the preset queue; and

[0026] comparing the acquired total number of surplus tokens with the number of lacked tokens.

[0027] Taking the next node of the token node in the preset queue as the new token node, includes:

[0028] pointing the second pointer to the new token node.

[0029] In a possible implementation manner, after storing the total number of surplus tokens into the node of the preset queue corresponding to the current statistical period, the method further includes:

[0030] in a case where the next node of the node corresponding to the current statistical period is a same node as the token node, taking the next node of the node corresponding to the current statistical period as a node corresponding to a new current statistical period, emptying the node corresponding to the new current statistical period, and taking the next node of the token node as the new token node.

[0031] According to another aspect of the present disclosure, a token management method is provided. The method is applied to a second token proxy module and includes:

[0032] acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods; wherein N is a positive integer;

[0033] updating a maximum basic token number of the second token proxy module according to the allocation numbers;

[0034] determining a number of lacked tokens according to the updated maximum basic token number; and

[0035] generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

[0036] In a possible implementation manner, updating the maximum basic token number of the second token proxy module according to the allocation numbers, includes:

[0037] in a case where at least one of the allocation numbers is greater than zero, taking a sum of a maximum allocation number and the maximum basic token number as the updated maximum basic token number; and/or

[0038] in a case where all the allocation numbers are zero, updating the maximum basic token number to a preset value.

[0039] According to another aspect of the present disclosure, an information sending method is provided and includes:

[0040] determining, by a second token proxy module, a number of lacked tokens according to a current maximum basic token number;

[0041] generating, by the second token proxy module, token demand information carrying the number of lacked tokens, and sending, by the second token proxy module, the token demand information to a first token proxy module;

[0042] determining, by the first token proxy module, the number of lacked tokens, in response to receiving the token demand information; and

[0043] acquiring, by the first token proxy module, surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating, by the first token proxy module, the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0044] According to another aspect of the present disclosure, a token management apparatus is provided. The apparatus is applied to a first token proxy module and includes:

[0045] a demand analysis unit, configured for determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module; and

[0046] a token allocation unit, configured for acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information; wherein the preset queue includes multiple nodes, one of the multiple nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period.

[0047] According to another aspect of the present disclosure, a token management apparatus is provided. The apparatus is applied to a second token proxy module and includes:

[0048] a data acquisition unit, configured for acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods; wherein N is a positive integer;

[0049] an update unit, configured for updating a maximum basic token number of the second token proxy module according to the allocation numbers;

[0050] a data processing unit, configured for determining a number of lacked tokens according to the updated maximum basic token number; and

[0051] a demand generation unit, configured for generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

[0052] According to another aspect of the present disclosure, an information sending apparatus is provided. The apparatus include: a first token proxy module and at least one second token proxy module; wherein,

[0053] the second token proxy module is configured for: determining a number of lacked tokens according to a current maximum basic token number, generating token demand information carrying the number of lacked tokens, and sending the token demand information to the first token proxy module; and

[0054] the first token proxy module is configured for: in response to receiving the token demand information, determining the number of lacked tokens, acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0055] According to another aspect of the present disclosure, an electronic device is provided and includes:

[0056] at least one processor; and

[0057] a memory communicatively connected to the at least one processor, wherein,

[0058] the memory stores instructions that are executable by the at least one processor, and the instructions, when executed by the at least one processor, enable the at least one processor to perform the method in any embodiment of the present disclosure.

[0059] According to another aspect of the present disclosure, a non-transitory computer readable storage medium storing computer instructions is provided. The computer instructions are configured for causing a computer to perform the method in any embodiment of the present disclosure.

[0060] According to another aspect of the present disclosure, a computer program product including a computer program/computer instructions is provided. The computer program/computer instructions, when performed by a processor, implements/implement the method in any embodiment of the present disclosure.

[0061] The present disclosure provides a token management method and apparatus, an information sending method and apparatus, an electronic device, and a non-transitory computer readable storage medium storing computer instructions. The first token proxy module determines the number of lacked tokens, in response to receiving token demand information sent by a second token proxy module, acquires surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocates the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information. The preset queue includes multiple nodes, one of the multiple nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period. The total number of surplus tokens of the respective second token proxy modules is stored in the preset queue, and when a second token proxy module has a token demand, tokens are scheduled for the second token proxy module from the total surplus tokens, effectively achieving multi-machine (token proxy modules) load balancing, improving the fairness and efficiency of token allocation, and helping to improve the efficiency of information sending.

[0062] It should be understood that the contents described in this section are not intended to identify key or important features of embodiments of the present disclosure, and also are not intended to limit the scope of the present disclosure. Other features of the present disclosure will become easily understood from the following description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0063] The drawings are used to better understand the solution and do not constitute a limitation to the present disclosure. In the drawings:

[0064] FIG. 1 is a first one of flowcharts of token management methods according to the present disclosure.

[0065] FIG. 2 is a first one of schematic diagrams of multi-producer-multi-consumer models according to the present disclosure.

[0066] FIG. 3 is a second one of schematic diagrams of multi-producer-multi-consumer models according to the present disclosure.

[0067] FIG. 4 is a schematic diagram of a token bucket according to the present disclosure.

[0068] FIG. 5 is a schematic diagram of a preset queue according to the present disclosure.

[0069] FIG. 6 is a second one of flowcharts of token management methods according to the present disclosure.

[0070] FIG. 7 is a schematic diagram of adjustment of a balance horizontal line according to the present disclosure.

[0071] FIG. 8 is a flowchart of an information sending method according to the present disclosure.

[0072] FIG. 9 is a first one of schematic structural diagrams of token management apparatuses according to the present disclosure.

[0073] FIG. 10 is a second one of schematic structural diagrams of token management apparatuses according to the present disclosure.

[0074] FIG. 11 is a schematic structural diagram of an information sending apparatus according to the present disclosure.

[0075] FIG. 12 is a schematic structural diagram of an electronic device according to the present disclosure.

DETAILED DESCRIPTION

[0076] The exemplary embodiments of the present disclosure will be described below in combination with drawings, including various details of the embodiments of the

present disclosure to facilitate understanding, which should be considered as exemplary only. Therefore, those of ordinary skill in the art should realize that various changes and modifications may be made to the embodiments described herein without departing from the scope and spirit of the present disclosure. Likewise, the description of well-known functions and structures are omitted in the following description for clarity and conciseness.

[0077] For the defect of the low efficiency in both the token scheduling and information sending due to the inability of traditional token allocation algorithms to achieve the multi-machine load balancing, the present disclosure provides a token management method and apparatus, an information sending method and apparatus, an electronic device, and a non-transitory computer readable storage medium storing computer instructions. In the present disclosure, a preset queue is provided in a first token proxy module, the preset queue includes multiple nodes, one of the multiple nodes is used to store the total number of surplus tokens corresponding to a statistical period, and the total number of the surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period. When the second token proxy module sends token demand information to the first token proxy module, the first token proxy module schedules tokens for the second token proxy module from the total surplus tokens, effectively achieving multi-machine load balancing, improving the fairness and efficiency of token allocation, thereby helping to improve the efficiency of information sending.

[0078] The token management method of the present disclosure is explained below through specific embodiments.

[0079] FIG. 1 shows a flowchart of a token management method of an embodiment of the present disclosure. The executive subject of the embodiment may be a terminal with computing power or a first token proxy module provided on the terminal. As shown in FIG. 1, the token management method of the embodiment of the present disclosure may include the following steps S110-S120.

[0080] At S110, determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module.

[0081] For example, each host runs a token proxy module. From multiple token proxy modules running on multiple hosts, a token proxy module is selected as the first token proxy module and the remaining token proxy modules are taken as the second

token proxy modules. As shown in FIG. 2, each host is provided with a local producer and a local consumer to maintain a local token bucket. The local producer places the produced tokens in the local token bucket, while the local consumer, for example, the data packet processed by the host, consumes tokens from the local token bucket. When there is no token in the local token bucket, the data packet is discarded or cached, to achieve a rate limiting effect. The above multiple hosts form a multi-producer-multi-consumer model.

[0082] As shown in FIG. 3, a second token proxy module reads and writes token data in the local token bucket, such as the number of surplus tokens, the number of lacked tokens, etc., and the second token proxy module communicates with the first token proxy module, to send the read token data to the first token proxy module. In some embodiments, the first token proxy module may also read token data from the local token bucket of the first token proxy module, and combine the token data read by itself with the token data sent by respective second token proxy modules, to calculate the following total number of surplus tokens.

[0083] The first token proxy module may be manually configured or elected from respective token proxy modules through a self-negotiation protocol. The second token proxy modules only communicate with the first token proxy module. In addition to communicating with the respective second token proxy modules, the first token proxy module is also responsible for maintaining additional token bucket, and this token bucket is used to store the following preset queue.

[0084] The second token proxy module counts the token data of the token bucket once per statistical period, and reports the counted number of surplus tokens, etc., to the first token proxy module. The above first token proxy module may be considered as a main token proxy module, and the second token proxy module may be considered as a slave token proxy module.

[0085] After the second token proxy module counts the token data, the second token proxy module may generate token demand information according to the number of lacked tokens and send the token demand information to the first token proxy module.

[0086] On the premise that a host has already got required tokens to forward information, the surplus tokens are remaining tokens of the host other than the above tokens required for forwarding information. This part of tokens can be provided for use

by other hosts. The lacked tokens refer to additional tokens required for a host to meet its requirements to forward information.

[0087] At S120, acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0088] The above second token proxy modules may also include the first token proxy module. That is, the first token proxy module may also count the number of surplus tokens of the first token proxy module into the total number of the overall surplus tokens. At the same time, the first token proxy module may also have the demand for sending information, and the processing manner is the same as that of the second token proxy module.

[0089] The preset queue includes multiple nodes, one of the multiple nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period. The respective nodes of the preset queue store the respective total numbers of surplus tokens in a chronological order.

[0090] In some embodiments, a balance horizontal line is provided for the token bucket. The value corresponding to the balance horizontal line is the number of tokens meeting supply-demand balance. The value is the number of tokens in the token bucket when token production is balanced with token consumption. The balance horizontal line is indicated by the reference number 41 in FIG. 4.

[0091] The tokens, in the token bucket, that exceed the number of tokens meeting the supply-demand balance are surplus tokens. As shown in FIG. 4, the surplus tokens 42 are tokens in the token area above the balance horizontal line. The maximum number of surplus tokens 43 corresponds to the upper limit of the capacity of the token bucket, and tokens exceeding the upper limit are directly discarded.

[0092] As shown in FIG. 4, tokens in the token area that is below the balance horizontal line and is above the zero line, are base tokens 44. The maximum basic token number 45 is the number of tokens meeting supply-demand balance corresponding to the balance horizontal line. The zero line represents the positive and negative boundary line for the number of tokens in the token bucket, and if the number of tokens in the token

bucket is lower than the boundary line, the suppression of consumption is triggered, such as discarding data packets or caching data packets.

[0093] As shown in FIG. 4, if the number of tokens in the token bucket is less than the number of tokens meeting supply-demand balance corresponding to the balance horizontal line, tokens corresponding to the empty token area from the basic tokens to the balance horizontal line are lacked tokens 46.

[0094] As shown in FIG. 4, the number of the lacked tokens 46 may be greater than the maximum base token number. At this time, the number of tokens in the token bucket may be negative, and lacked tokens below the zero line are taken as negative tokens 47, that is, tokens in the empty token area below the zero line are taken as the negative tokens. The state with a negative token is generally caused by burst traffic or special scenarios. Because the token consumption is suppressed in this state, this state will not be maintained for a long time.

[0095] In some embodiments, the first token proxy module may store or update the total number of surplus tokens according to the following steps:

[0096] in response to receiving the number of surplus tokens reported by the at least one second token proxy module, acquiring a total number of surplus tokens from a node corresponding to a current statistical period, updating the total number of surplus tokens corresponding to the current statistical period according to the at least one received number of surplus tokens, and using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period.

[0097] In a practical application, the respective second token proxy modules may synchronously upload the numbers of surplus tokens or asynchronously upload the numbers of surplus tokens, which is not limited in the present disclosure. In the case where the respective second token proxy modules asynchronously upload the numbers of surplus tokens, a certain node in the preset queue stores the accumulated value of the numbers of surplus tokens, that is, the value obtained by accumulating the numbers of surplus tokens asynchronously uploaded by the respective second token proxy modules, in the statistical period corresponding to the node.

[0098] In some embodiments, the preset queue includes a circular queue, as shown in FIG. 5. A first pointer is provided on the preset queue, and the first pointer is used for pointing to the node, corresponding to the current statistical period, in the preset

queue. When storing or updating the above total number of surplus tokens, a first position of the first pointer in the preset queue is acquired, and according to the first position, the total number of surplus tokens is acquired from the node corresponding to the current statistical period; the total number of surplus tokens corresponding to the current statistical period is updated according to the received at least one number of surplus tokens; and the node corresponding to the current statistical period is emptied according to the first position, and the updated total number of surplus tokens is stored into the node corresponding to the current statistical period. At the end of the current statistical period, according to a preset direction, the first pointer is pointed to a next node of the node corresponding to the current statistical period, and the new node currently pointed to by the first pointer is emptied. The preset direction may be a clockwise direction or a counterclockwise direction, which will not be limited in the present disclosure.

[0099] As shown in FIG. 5, the preset queue may include multiple nodes, for example N nodes, where N is a positive integer. For example, N takes a value of 8. By the preset queue, the total numbers of surplus tokens reported in the recent period may be saved, traffic bursts in N statistical periods are maintained, and token data is received from respective main token proxy modules, for example, the number of surplus tokens, and reward tokens are fed back when there is a token demand, that is, tokens are allocated from the overall surplus tokens to the token proxy module with token demand. Using a circular queue to store the total number of surplus tokens in N statistical periods can prevent the trend of infinite accumulation of tokens, and retain only the surplus tokens in the recent few statistical periods.

[0100] As shown in FIG. 5, a second pointer is provided on the preset queue, the second pointer is used for pointing to a token node in the preset queue, and the token node is a non-empty node with an earliest time in the preset queue. When the total number of surplus tokens in the node is zero, the node is emptied. The first pointer points to the tail of the preset queue, and the second pointer points to the head of the preset queue.

[0101] When the second token proxy module counts that the local token bucket is in a lacked token state, the second token proxy module generates the token demand information according to the number of lacked tokens and sends the token demand information to the first token proxy module. The first token proxy module compares the

number of lacked tokens carried in the token demand information with the total number of surplus tokens stored in the token node; in a case where the total number of surplus tokens stored in the token node is less than the number of lacked tokens, the first token proxy module extracts all the surplus tokens from the token node, empties the token node, and updates the number of lacked tokens according to the number of the extracted surplus tokens, that is, calculating the number of lacked tokens minus the number of the surplus tokens extracted from the token node, and taking the calculated difference value as the updated number of lacked tokens. Then, a next node of the token node in the preset queue is taken as a new token node, and the above step of comparing the total number of surplus tokens stored in the token node with the number of lacked tokens is returned, until the total number of the extracted surplus tokens is equal to the number of lacked tokens carried in the token demand information.

[0102] In a case where the total number of surplus tokens stored in the token node is larger than the number of lacked tokens, surplus tokens of a same number as the number of lacked tokens are extracted from the token node, and the total number of surplus tokens stored in the token node is updated.

[0103] In a case where the total number of surplus tokens stored in the token node is equal to the number of lacked tokens, the surplus tokens are extracted from the token node, the token node is emptied, and the next node of the token node in the preset queue is taken as the new token node.

[0104] In summary, when the first token proxy module receives the token demand information, the first token proxy module extracts the surplus tokens from the token node in the preset queue, to meet the token demand of the second token proxy module. If the current token node is emptied and the token demand of the second token proxy module is still not met, the surplus token(s) is acquired from the next node of the token node according to the preset direction, to meet the token demand of the second token proxy module. When the surplus token(s) is acquired from a new token node, it is necessary to first determine whether the new token node is also the node corresponding to the current statistical period. If the new token node is not the node corresponding to the current statistical period, the step of acquiring the surplus token(s) from the new token node to meet the token demand of the second token proxy module is performed.

[0105] The token scheduling manner in the above embodiments can solve the efficiency issue and fairness issue of token scheduling algorithms in multi-core and high-traffic scenario.

[0106] In some embodiments, comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens may be achieved by using the following steps:

[0107] first, acquiring a second position of a second pointer in the preset queue; wherein the second pointer is used for pointing to the token node in the preset queue; then, acquiring, according to the second position, the total number of surplus tokens stored in the token node from the preset queue; and finally, comparing the acquired total number of surplus tokens with the number of lacked tokens.

[0108] The next node of the token node in the preset queue is taken as a new token node, and the second pointer is pointed to the new token node.

[0109] After the total number of surplus tokens is stored into the node in the preset queue corresponding to the current statistical period, if the next node of the node corresponding to the current statistical period is a same node as the token node, the next node of the node corresponding to the current statistical period is taken as a node corresponding to a new current statistical period, and the node corresponding to the new current statistical period is emptied; and the next node of the token node is taken as the new token node.

[0110] A method executed by the second token proxy module in the present disclosure is stated below through specific embodiments. As shown in FIG. 6, a token management method executed by the second token proxy module may include the following steps S610-S640.

[0111] At S610, acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods.

[0112] When the second token proxy module needs a token(s), the second token proxy module generates token demand information and sends the token demand information to the first token proxy module. The first token proxy module allocates surplus tokens to the second token proxy module, according to the number of lacked tokens number carried in the token demand information. This step is to acquire the

allocation numbers of the surplus tokens allocated by the first token proxy module to the second token proxy module in the recent N allocation periods.

[0113] The above allocation period may be the same as or may be different from the above statistical period, which will not be limited in the present disclosure.

[0114] At S620, updating a maximum basic token number of the second token proxy module according to the allocation numbers. At the same time, the allocated tokens are updated to the local token bucket of the second token proxy module.

[0115] Firstly, the current maximum basic token number is acquired, and then the current maximum basic token number is updated according to the allocation numbers. According to the numbers of surplus tokens acquired in the recent N allocation periods, that is, according to the condition of insufficient tokens in the recent N allocation periods, a new maximum basic token number that better meets the current traffic demand can be determined.

[0116] Exemplarily, the value of N may be flexibly set according to the requirements of the actual application scenario, for example, the value is set to 4.

[0117] At S630, determining a number of lacked tokens according to the updated maximum basic token number.

[0118] At S640, generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

[0119] In some embodiments, updating the maximum basic token number of the second token proxy module according to the allocation numbers may be achieved by using the following steps.

[0120] In a case where at least one of the allocation numbers is greater than zero, that is, in a case where surplus tokens are allocated to the second token proxy module in the recent N allocation periods, a sum of a maximum allocation number and the maximum basic token number is taken as the updated maximum basic token number.

[0121] In a case where all the allocation numbers are zero, the maximum basic token number is updated to a preset value. If no token has been allocated in the recent N allocation periods, it indicates that the local demand is decreased or the total number of surplus tokens is zero. At this point, the size (balance horizontal line) of the token bucket

is restored to the initial value, that is, the preset value, i.e. the maximum basic token number.

[0122] In order to make the number of lacked tokens closer to the actual demand for traffic, a method for dynamically adjusting the size (balance horizontal line) of the token bucket is introduced. The basic principle is to reflect the locally allocated tokens into a demand reported next time, such that more tokens can be allocated next time. Considering fairness, the maximum basic token number is not set to a large value at the beginning. Respective local token buckets apply for it from a basic value, and then quickly converge and increase if the total number of surplus tokens is large.

[0123] As shown in FIG. 7, in a state at which the size of the token bucket has not changed, the balance horizontal line 71 is at the initial preset position; the balance horizontal line 72 increases according to at least one recent allocation number; and the balance horizontal line 73 returns to the original position because no token has been allocated in the recent N allocation periods.

[0124] The number of lacked tokens (close to the actual traffic demand) reported according to the above scheme is determined according to the number of surplus tokens allocated in the recent N allocation periods. When the number of total surplus tokens is large, the local demand may rapidly increase while the fairness of the token allocation can be improved.

[0125] Correspondingly to the above token management method, the present disclosure also provides an information sending method. As shown in FIG. 8, the method may include the following steps S810-S840.

[0126] At S810, determining, by a second token proxy module, a number of lacked tokens according to a current maximum basic token number.

[0127] At S820, generating, by the second token proxy module, token demand information carrying the number of lacked tokens, and sending, by the second token proxy module, the token demand information to a first token proxy module.

[0128] At S830, determining, by the first token proxy module, the number of lacked tokens, in response to receiving the token demand information.

[0129] At S840, acquiring, by the first token proxy module, surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating, by the first token proxy module, the acquired surplus tokens to the second

token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0130] The first token proxy module and second token proxy module in this embodiment are the same as the first token proxy module and second token proxy module in the above embodiments, and the duplicate content will not be repeated.

[0131] The information sending method in this embodiment may be used as a rate limiting method. In a practical application, the method may be deployed as a Linux kernel module and loaded into the kernel through insmod, and the three-layer data packets (IP messages) are hooked through the kernel netfilter mechanism. In a kernel mode, the local token bucket is maintained, while the token proxy module acts as a user mode program, interacts with the token bucket in kernel mode, and then reports the token data or token demand information to the first token proxy module.

[0132] Based on the same invention concept, a token management apparatus corresponding to the token management method is also provided in an embodiment of the present disclosure. Since the principle of solving problems by the apparatus in the embodiment of the present disclosure is similar to the above token management method in the embodiment of the present disclosure, the implementation of the apparatus may be referred to the implementation of the method, and the duplicate content will not be repeated.

[0133] As shown in FIG. 9, FIG. 9 is a schematic structural diagram of a token management apparatus provided by an embodiment of the present disclosure. The apparatus is applied to a first token proxy module and includes:

[0134] a demand analysis unit 910, configured for determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module; and

[0135] a token allocation unit 920, configured for acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0136] The preset queue includes multiple nodes, one of the multiple nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of the number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period.

[0137] In some embodiments, the above apparatus further includes a data storage unit 930, configured for: in response to receiving the number of surplus tokens reported by the at least one second token proxy module, acquiring a total number of surplus tokens from a node corresponding to a current statistical period, updating the total number of surplus tokens corresponding to the current statistical period according to the at least one received number of surplus tokens, and using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period; wherein respective nodes of the preset queue store respective total numbers of surplus tokens in a chronological order.

[0138] In some embodiments, the data storage unit 930, when acquiring the total number of surplus tokens from the node corresponding to the current statistical period, is configured for:

[0139] acquiring a first position of a first pointer in the preset queue; wherein the first pointer is used for pointing to the node, corresponding to the current statistical period, in the preset queue; and

[0140] acquiring, according to the first position, the total number of surplus tokens from the node corresponding to the current statistical period.

[0141] The data storage unit 930, when using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period, is configured for:

[0142] emptying the node corresponding to the current statistical period according to the first position, and storing the updated total number of surplus tokens into the node corresponding to the current statistical period; and

[0143] at the end of the current statistical period, pointing, according to a preset direction, the first pointer to a next node of the node corresponding to the current statistical period; wherein the preset queue includes a circular queue.

[0144] In some embodiments, the token allocation unit 920, when acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, is configured for:

[0145] comparing a total number of surplus tokens stored in a token node of the preset queue with the number of lacked tokens; wherein the token node is a non-empty node with an earliest time in the preset queue;

[0146] in a case where the total number of surplus tokens stored in the token node is less than the number of lacked tokens, extracting the surplus tokens from the token node, and updating the number of lacked tokens according to the number of the extracted surplus tokens; and

[0147] taking a next node of the token node in the preset queue as a new token node, and returning the step of comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens, until a total number of the extracted surplus tokens is equal to the number of lacked tokens carried in the token demand information.

[0148] In some embodiments, the token allocation unit 920, when acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, is further configured for:

[0149] in a case where the total number of surplus tokens stored in the token node is larger than the number of lacked tokens, extracting surplus tokens of a same number as the number of lacked tokens from the token node, and updating the total number of surplus tokens stored in the token node; and/or

[0150] in a case where the total number of surplus tokens stored in the token node is equal to the number of lacked tokens, extracting the surplus tokens from the token node, emptying the token node, and taking the next node of the token node in the preset queue as the new token node.

[0151] In some embodiments, the token allocation unit 920, when comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens, is configured for:

[0152] acquiring a second position of a second pointer in the preset queue; wherein the second pointer is used for pointing to the token node in the preset queue;

[0153] acquiring, according to the second position, the total number of surplus tokens stored in the token node from the preset queue; and

[0154] comparing the acquired total number of surplus tokens with the number of lacked tokens; and

[0155] taking the next node of the token node in the preset queue as the new token node, includes:

[0156] pointing the second pointer to the new token node.

[0157] In some embodiments, the data storage unit 930, after storing the total number of surplus tokens into the node of the preset queue corresponding to the current statistical period, is further configured for:

[0158] in a case where the next node of the node corresponding to the current statistical period is a same node as the token node, taking the next node of the node corresponding to the current statistical period as a node corresponding to a new current statistical period, emptying the node corresponding to the new current statistical period, and taking the next node of the token node as the new token node.

[0159] Based on the same invention concept, a token management apparatus corresponding to the token management method is also provided in an embodiment of the present disclosure. Since the principle of solving problems by the apparatus in the embodiment of the present disclosure is similar to the above token management method in the embodiment of the present disclosure, the implementation of the apparatus may be referred to the implementation of the method, and the duplicate content will not be repeated.

[0160] As shown in FIG. 10, FIG. 10 is a schematic structural diagram of a token management apparatus provided by an embodiment of the present disclosure. The apparatus is applied to a second token proxy module and includes:

[0161] a data acquisition unit 1010, configured for acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods; wherein N is a positive integer;

[0162] an update unit 1020, configured for updating a maximum basic token number of the second token proxy module according to the allocation numbers;

[0163] a data processing unit 1030, configured for determining a number of lacked tokens according to the updated maximum basic token number; and

[0164] a demand generation unit 1040, configured for generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

[0165] In some embodiments, the update unit 1020, when updating the maximum basic token number of the second token proxy module according to the allocation numbers, is configured for:

[0166] in a case where at least one of the allocation numbers is greater than zero, taking a sum of a maximum allocation number and the maximum basic token number as the updated maximum basic token number; and/or

[0167] in a case where all the allocation numbers are zero, updating the maximum basic token number to a preset value.

[0168] Based on the same invention concept, an information sending apparatus corresponding to the information sending method is also provided in an embodiment of the present disclosure. Since the principle of solving problems by the apparatus in the embodiment of the present disclosure is similar to the above information sending method in the embodiment of the present disclosure, the implementation of the apparatus may be referred to the implementation of the method, and the duplicate content will not be repeated.

[0169] As shown in FIG. 11, FIG. 11 is a schematic structural diagram of an information sending apparatus provided by an embodiment of the present disclosure. The apparatus includes a first token proxy module 1110 and at least one second token proxy module 1120.

[0170] The second token proxy module 1120 is configured for: determining a number of lacked tokens according to a current maximum basic token number, generating token demand information carrying the number of lacked tokens, and sending the token demand information to the first token proxy module.

[0171] The first token proxy module 1110 is configured for: in response to receiving the token demand information, determining the number of lacked tokens, acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

[0172] In the technical solutions of the present disclosure, the obtaining, storage, application, etc., of the user's personal information involved comply with the provisions of relevant laws and regulations, and do not violate public order and moral.

[0173] According to embodiments of the present disclosure, the present disclosure also provides an electronic device, a readable storage medium, and a computer program product.

[0174] FIG. 12 shows a schematic block diagram of an exemplary electronic device 1200 that may be used to implement the embodiments of the present disclosure. The electronic device is intended to represent various forms of digital computers, such as laptop computers, desktop computers, workbenches, personal digital assistants, servers, blade servers, mainframe computers, and other suitable computers. The electronic device may also represent various forms of mobile apparatuses, such as personal digital assistants, cellular phones, smart phones, wearable devices, and other similar computing apparatuses. The components, their connections and relationships, and their functions shown herein are merely examples, and are not intended to limit the implementation of the present disclosure described and/or claimed herein.

[0175] As shown in FIG. 12, the device 1200 includes a computing unit 1210 that may perform various appropriate actions and processes according to a computer program stored in a read only memory (ROM) 1220 or a computer program loaded from the storage unit 1280 into a random access memory (RAM) 1230. In the RAM 1230, various programs and data required for the operation of the device 1200 may also be stored. The computing unit 1210, the ROM 1220, and the RAM 1230 are connected to each other through a bus 1240. The input/output (I/O) interface 1250 is also connected to the bus 1240.

[0176] Multiple components in the device 1200 are connected to the I/O interface 1250, including: an input unit 1260, for example, a keyboard, a mouse, etc.; an output unit 1270, for example, various types of displays, speakers, etc.; a storage unit 1280, for example, a magnetic disk, an optical disk, etc.; and a communication unit 1290, for example, a network card, a modem, a wireless communication transceiver, etc. The communication unit 1290 allows the device 1200 to exchange information/data with other devices through computer networks such as the Internet and/or various telecommunication networks.

[0177] The computing unit 1210 may be various general-purpose and/or special-purpose processing components with processing and computing capabilities. Some examples of the computing unit 1210 include, but are not limited to, a central processing unit (CPU), a graphics processing unit (GPU), various special-purpose artificial intelligence (AI) computing chips, various computing units running machine learning model algorithms, digital signal processors (DSP), and any appropriate processors, controllers, microcontrollers, etc. The computing unit 1210 performs various methods

and processes described above, for example, the token management method or the information sending method. For example, in some embodiments, the token management method or the information sending method may be implemented as a computer software program that is tangibly contained in a machine readable medium, for example, the storage unit 1280. In some embodiments, part or all of the computer program may be loaded and/or installed on the device 1200 via the ROM 1220 and/or the communication unit 1290. When the computer program is loaded into the RAM 1230 and performed by computing unit 1210, one or more steps of the token management method or the information sending method described above may be performed. Optionally, in other embodiments, the computing unit 1210 may be configured for performing the token management method or the information sending method via any other suitable means (for example, by means of firmware).

[0178] Various implementation manners of the systems and technologies described above herein may be implemented in a digital electronic circuit system, an integrated circuit system, a field programmable gate array (FPGA), an application specific integrated circuit (ASIC), an application special standard product (ASSP), a system on chip (SOC), a complex programmable logic device (CPLD), a computer hardware, firmware, software and/or combinations thereof. These various implementation manners may include: implementations in one or more computer programs. The one or more computer programs may be executed and/or interpreted on a programmable system that includes at least one programmable processor. The programmable processor may be a special-purpose or general-purpose programmable processor, and may receive data and instructions from a storage system, at least one input apparatus, and at least one output apparatus, and transmit the data and instructions to the storage system, the at least one input apparatus, and the at least one output apparatus.

[0179] The program codes for implementing the method of the present disclosure may be written in any combination of one or more programming languages. These program codes may be provided to the processor or controller of a general-purpose computer, a special-purpose computer or other programmable data processing apparatuses, such that the program codes, when executed by the processor or controller, enables the functions/operations specified in the flowchart and/or block diagram to be implemented. The program codes may be executed completely on the machine, partially

on the machine, partially on the machine and partially on the remote machine as a separate software package, or completely on the remote machine or server.

[0180] In the context of the present disclosure, a machine readable medium may be a tangible medium that may contain or store programs for use by or in combination with an instruction execution system, apparatus, or device. The machine readable medium may be a machine readable signal medium or a machine readable storage medium. The machine readable medium may include, but is not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus or device, or any suitable combination of the above contents. A more specific example of the machine readable storage medium will include an electrical connection based on one or more lines, a portable computer disk, a hard disk, a random access memory (RAM), a read only memory (ROM), an erasable programmable read only memory (EPROM or flash memory), an optical fiber, a portable compact disk read only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the above contents.

[0181] In order to provide interactions with a user, the system and technology described herein may be implemented on a computer. The computer has: a display apparatus (for example, a CRT (cathode ray tube) or an LCD (liquid crystal display) monitor) for displaying information to the user; and a keyboard and a pointing apparatus (for example, a mouse or a trackball), through which the user may provide input to the computer. Other kinds of apparatuses may also be used to provide interactions with a user; for example, the feedback provided to a user may be any form of sensory feedback (for example, visual feedback, auditory feedback, or tactile feedback); and input from a user may be received by using any form (including acoustic input, voice input, or tactile input).

[0182] The systems and techniques described herein may be implemented in a computing system (for example, as a data server) that includes back-end components, or be implemented in a computing system (for example, an application server) that includes middleware components, or be implemented in a computing system (for example, a user computer with a graphical user interface or a web browser through which the user may interact with the implementation manner of the systems and technologies described herein) that includes front-end components, or be implemented in a computing system that includes any combination of such back-end components, intermediate components,

or front-end components. The components of the system may be interconnected by any form or medium of digital data communication (for example, a communication network). The example of the communication network includes a local area network (LAN), a wide area network (WAN), and the Internet.

[0183] The computer system may include a client and a server. The client and the server are generally remote from each other and typically interact through a communication network. The client-server relationship is generated by computer programs that run on respective computers and have a client-server relationship with each other. The server may be a cloud server, a server of a distributed system, or a server that combines blockchain.

[0184] It should be understood that various forms of processes shown above may be used to reorder, add, or delete steps. For example, respective steps recited in the present disclosure may be executed in parallel, or may be executed sequentially, or may be executed in a different order, as long as the desired result of the technical solution disclosed in the present disclosure can be achieved, which is not limited herein.

[0185] The above specific implementation manners do not constitute a limitation on the protection scope of the present disclosure. It should be understood by those skilled in the art that various modifications, combinations, sub-combinations, and substitutions may be made according to design requirements and other factors. Any modification, equivalent replacement, and improvement, and the like made within the spirit and principle of the present disclosure shall be contained in the protection scope of the present disclosure.

CLAIMS

What is claimed is:

1. A token management method, applied to a first token proxy module, comprising:

determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module; and

acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information;

wherein the preset queue comprises a plurality of nodes, one of the plurality of nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of a number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period.

2. The method of claim 1, further comprising:

in response to receiving the number of surplus tokens reported by the at least one second token proxy module, acquiring a total number of surplus tokens from a node corresponding to a current statistical period, updating the total number of surplus tokens corresponding to the current statistical period according to the at least one received number of surplus tokens, and using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period;

wherein respective nodes of the preset queue store respective total numbers of surplus tokens in a chronological order.

3. The method of claim 2, wherein the acquiring the total number of surplus tokens from the node corresponding to the current statistical period, comprises:

acquiring a first position of a first pointer in the preset queue; wherein the first pointer is used for pointing to the node, corresponding to the current statistical period, in the preset queue; and

acquiring, according to the first position, the total number of surplus tokens from the node corresponding to the current statistical period; and

the using the updated total number of surplus tokens to update the total number of surplus tokens stored in the node corresponding to the current statistical period, comprises:

emptying the node corresponding to the current statistical period according to the first position, and storing the updated total number of surplus tokens into the node corresponding to the current statistical period; and

at the end of the current statistical period, pointing, according to a preset direction, the first pointer to a next node of the node corresponding to the current statistical period, and emptying the node currently pointed to by the first pointer; wherein the preset queue comprises a circular queue.

4. The method of claim 2 or 3, wherein the acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, comprises:

comparing a total number of surplus tokens stored in a token node of the preset queue with the number of lacked tokens; wherein the token node is a non-empty node with an earliest time in the preset queue;

in a case where the total number of surplus tokens stored in the token node is less than the number of lacked tokens, extracting the surplus tokens from the token node, and updating the number of lacked tokens according to a number of the extracted surplus tokens; and

taking a next node of the token node in the preset queue as a new token node, and returning the step of comparing the total number of surplus tokens stored in the token

node of the preset queue with the number of lacked tokens, until a total number of the extracted surplus tokens is equal to the number of lacked tokens carried in the token demand information.

5. The method of claim 4, wherein the acquiring the surplus tokens from the at least one node of the preset queue according to the number of lacked tokens, further comprises:

in a case where the total number of surplus tokens stored in the token node is larger than the number of lacked tokens, extracting surplus tokens of a same number as the number of lacked tokens from the token node, and updating the total number of surplus tokens stored in the token node; and/or

in a case where the total number of surplus tokens stored in the token node is equal to the number of lacked tokens, extracting the surplus tokens from the token node, emptying the token node, and taking the next node of the token node in the preset queue as the new token node.

6. The method of claim 4, wherein the comparing the total number of surplus tokens stored in the token node of the preset queue with the number of lacked tokens, comprises:

acquiring a second position of a second pointer in the preset queue; wherein the second pointer is used for pointing to the token node in the preset queue;

acquiring, according to the second position, the total number of surplus tokens stored in the token node from the preset queue; and

comparing the acquired total number of surplus tokens with the number of lacked tokens;

the taking the next node of the token node in the preset queue as the new token node, comprises:

pointing the second pointer to the new token node.

7. The method of claim 4, further comprising:

in a case where the next node of the node corresponding to the current statistical period is a same node as the token node, taking the next node of the node corresponding to the current statistical period as a node corresponding to a new current statistical period, emptying the node corresponding to the new current statistical period, and taking the next node of the token node as the new token node.

8. A token management method, applied to a second token proxy module, comprising:

acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods; wherein N is a positive integer;

updating a maximum basic token number of the second token proxy module according to the allocation numbers;

determining a number of lacked tokens according to the updated maximum basic token number; and

generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

9. The method of claim 8, wherein the updating the maximum basic token number of the second token proxy module according to the allocation numbers, comprises:

in a case where at least one of the allocation numbers is greater than zero, taking a sum of a maximum allocation number and the maximum basic token number as the updated maximum basic token number; and/or

in a case where all the allocation numbers are zero, updating the maximum basic token number to a preset value.

10. An information sending method, comprising:

determining, by a second token proxy module, a number of lacked tokens according to a current maximum basic token number;

generating, by the second token proxy module, token demand information carrying the number of lacked tokens, and sending, by the second token proxy module, the token demand information to a first token proxy module;

determining, by the first token proxy module, the number of lacked tokens, in response to receiving the token demand information; and

acquiring, by the first token proxy module, surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating, by the first token proxy module, the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

11. A token management apparatus, applied to a first token proxy module, comprising:

a demand analysis unit, configured for determining a number of lacked tokens, in response to receiving token demand information sent by a second token proxy module; and

a token allocation unit, configured for acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information;

wherein the preset queue comprises a plurality of nodes, one of the plurality of nodes is used to store a total number of surplus tokens corresponding to a statistical period, and the total number of surplus tokens is a sum of a number of surplus tokens reported by at least one second token proxy module in the corresponding statistical period.

12. A token management apparatus, applied to a second token proxy module, comprising:

a data acquisition unit, configured for acquiring an allocation number of surplus tokens allocated by a first token proxy module to the second token proxy module in each of recent N allocation periods; wherein N is a positive integer;

an update unit, configured for updating a maximum basic token number of the second token proxy module according to the allocation numbers;

a data processing unit, configured for determining a number of lacked tokens according to the updated maximum basic token number; and

a demand generation unit, configured for generating token demand information carrying the number of lacked tokens and sending the token demand information to the first token proxy module, such that the first token proxy module allocates surplus tokens to the second token proxy module according to the token demand information.

13. An information sending apparatus, comprising a first token proxy module and at least one second token proxy module; wherein,

the second token proxy module is configured for: determining a number of lacked tokens according to a current maximum basic token number, generating token demand information carrying the number of lacked tokens, and sending the token demand information to the first token proxy module; and

the first token proxy module is configured for: in response to receiving the token demand information, determining the number of lacked tokens, acquiring surplus tokens from at least one node of a preset queue according to the number of lacked tokens, and allocating the acquired surplus tokens to the second token proxy module, such that the second token proxy module uses the allocated tokens to send information.

14. An electronic device, comprising:

at least one processor; and

a memory communicatively connected to the at least one processor, wherein,

the memory stores instructions that are executable by the at least one processor, and the instructions, when executed by the at least one processor, enable the at least one processor to perform the method of any one of claims 1-10.

15. A non-transitory computer readable storage medium storing computer instructions, wherein the computer instructions are configured for causing a computer to perform the method of any one of claims 1-10.